CR 54629

[redacted]

[redacted]

GENERAL DYNAMICS

Convair Division

[redacted]

A2136-1 (REV. 5-65)

GPO PRICE $ _____

CFSTI PRICE(S) $ _____

Hard copy (HC) 3.00

Microfiche (MF) .75

ff 653 July 65

SHADOW CONSTRAINT PROGRAM

GD|C-BTD65-041

30 June 1965

Contract NAS3-3232

R. N. Setter
Systems Technology, 591-0

Approved by R. Wentink
Assistant Chief Engineer
Centaur Design Analysis

GENERAL DYNAMICS | CONVAIR

ABSTRACT

The Shadow Constraint Program computes the positions and times of a space vehicle's entry into, and exit from, the shadow cones caused by interfering bodies (earth, moon, and planets) coming between the vehicle and the sun. The program is designed specifically for Centaur to determine if the time in the shadow for a particular lunar trajectory violates a constraint caused by the vehicle's design.

The program is divided into two subroutines: SHAD1 and SHAD2. SHAD1 is used with a numerical trajectory computing program and is incorporated in the GD/C N-Body Space Program. SHAD2 is used with an analytical (two-body) trajectory program and is used in the Lunar Targeting Program II.

The Shadow Constraint Program is written in FORTRAN for the IBM 7090/7094 computers. It was developed and coded by the Systems Technology group, 591-0, to specifications set by the Aeroballistics section of the Centaur project.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

SECTION 1
INTRODUCTION

The Shadow Constraint Program computes the trajectory's intersection with the shadow cone (umbra) as shown in Figure 1-1. The program is divided into two routines: one computes the shadow intersection with an analytical (two-body) trajectory, the other with an integrated (n-body) trajectory.

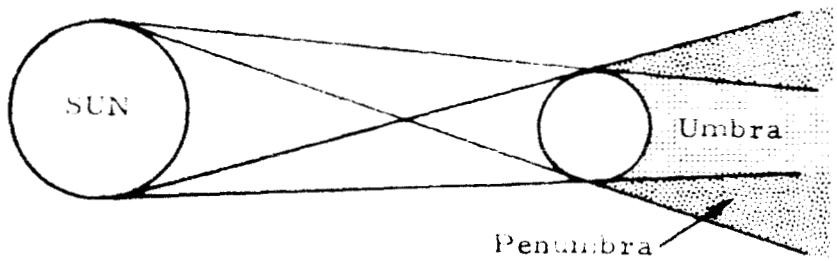

Figure 1-1. The Shadow Cone

With the n-body program the trajectory is computed by taking integration time steps. The shadow computation is simplified with this program by deciding (at each time point) whether the vehicle is in or out of the cone of darkness and interpolating for the shadow edge when the decision has been made that the vehicle has crossed the boundary from daylight to darkness or vice versa.

The high-order hyperosculatory interpolation formulas are used to find the edge of the shadow between the two integrated positions. Computation of the entry and exit points to the shadow is more difficult when computed from an analytical (two-body) trajectory. In this case, it is necessary to compute the intersection of the trajectory plane and the shadow cone. This is done by first projecting the sun-to-central body vector onto the trajectory plane and then computing the chord of the circle that is formed by slicing the cone at the vehicle's position vector. This gives an excellent approximation of the entry and exit points to the shadow. A refinement is then made through iteration to find the exact intersection.

SECTION 2

SHADOW COMPUTATIONS

This section presents the equations used to compute the trajectory's intersection with the shadow. Section 2.1 explains the shadow computation with the integrated (n-body) trajectory (SHAD1). Section 2.2 explains the computation with an analytically computed (two-body) trajectory (SHAD2).

2.1 SHADOW COMPUTATION WITH INTEGRATED TRAJECTORY. The shadow computation with the n-body program consists of comparing the radius of the shadow cone with the vehicle's distance from the center of the cone to determine whether the vehicle is in or out of the shadow. This is done at each integration step along the trajectory.

The radius of the shadow cone, B (Figure 2-1), is first computed as follows.

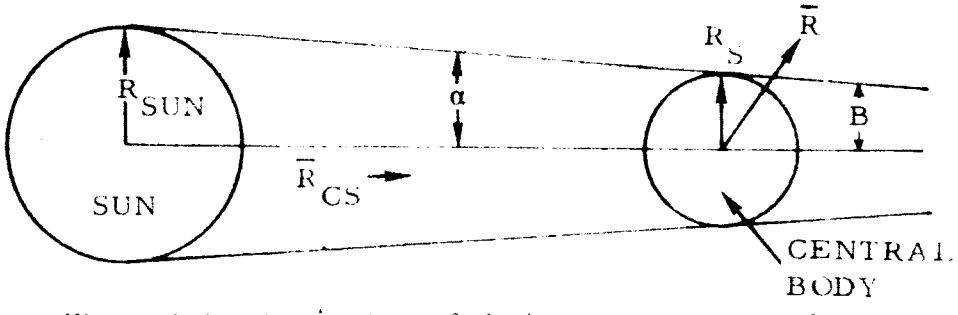

Figure 2-1. Comparison of Shadow Cone with Vehicle Distance

$$B = R_S - (\bar{R} \cdot \bar{1}R_{CS}) \tan \alpha$$

where

$\bar{R}$ = position of vehicle relative to central body

$\bar{R}_{CS}$ = position of central body relative to the sun

R_S = radius of central body at edge of shadow

$$\tan \alpha = \frac{R_{SUN} - R_{CB}}{|\bar{R}_{CS}|}$$

where

R_{SUN} = radius of sun

R_{CB} = radius of central body

NOTE: When the central body (C_B) is the earth (Figure 2-2), R_S is computed from the equation for an oblate spheroid as follows:

$$R_S = ab/\sqrt{a^2 + (a^2 - b^2)\sin \phi_T}$$

where

a = semimajor axis of the earth, and

b = the semiminor axis of the earth.

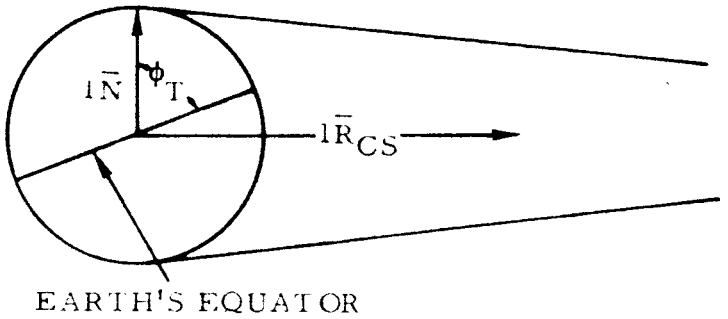

Figure 2-2. Radius of Earth Shadow Cone

$$\phi_T = \tan^{-1}\left(\bar{1}N(3)/\sqrt{\bar{1}N(1)^2 + \bar{1}N(2)^2}\right)$$

where

$$\bar{1}N = \bar{1}R_{CS} \times \left(\frac{\bar{R} \times \bar{R}_{CS}}{|\bar{R} \times \bar{R}_{CS}|}\right) ,$$

where $\bar{1}N$ is expressed in earth equatorial coordinates.

The vehicle's distance from the center of the shadow cone, R_C, (Figure 2-3), is computed as follows.

$$R_C = |\bar{R} \times \bar{1}R_{CS}|$$

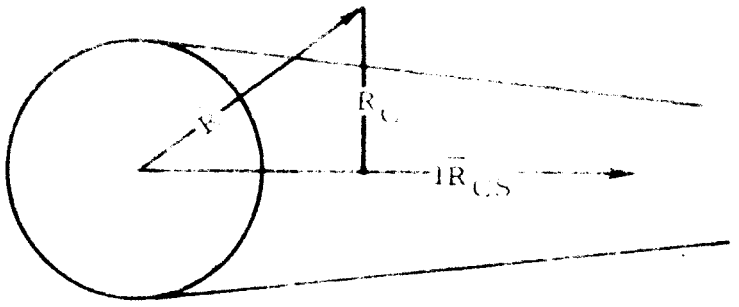

Figure 2-3. Vehicle Distance From Center of Shadow Cone

At each integration step, B is compared with R_C. When R_C > B, the vehicle is in the shadow, provided it is not on the daylight side of the body; i.e., $\bar{R} \cdot \bar{R}_{CS} < 0$ (Figure 2-4).

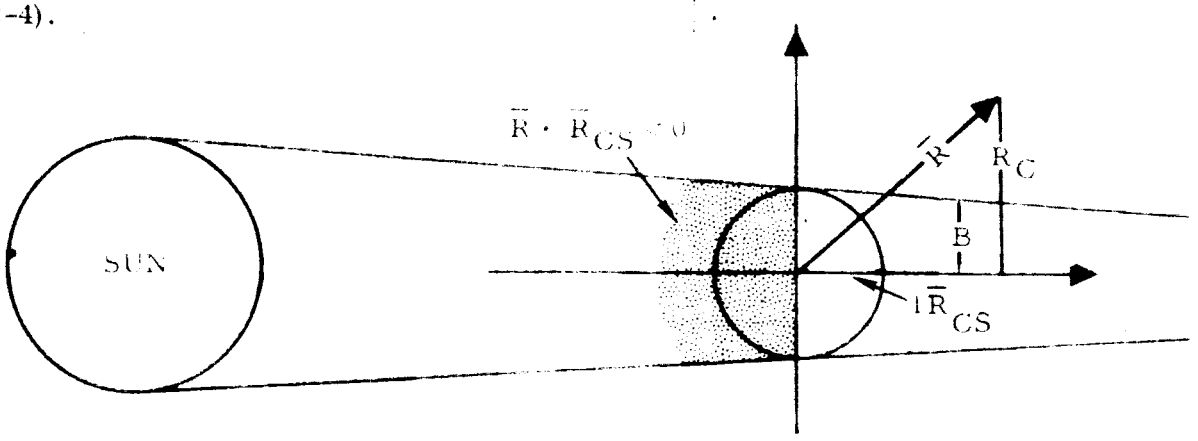

Figure 2-4. Comparison of B with R_C

When the boundary between daylight and darkness is crossed (determined by R_C - B changing sign), an iteration solving for the R_C - B = 0 condition is used to compute the exact boundary. (See Program Subroutines, subroutine INTRP.) The vehicle's position is computed between the two integration points using the two-point hyperosculatory interpolation formulas.

2.2 SHADOW COMPUTATION WITH TWO-BODY TRAJECTORY. The method used for shadow computation with a two-body trajectory is first to compute a good approximation of the intersection of the trajectory and the shadow cone and then to refine the computed intersection through iteration.

The intersection of the trajectory plane (Figure 2-5) and the plane perpendicular to the trajectory through the center of the shadow cone is obtained as follows.

$$\bar{h} = \bar{R}_{INJ} \times \bar{V}_{INJ}$$

$$\bar{1}C = \frac{\bar{1}R_{CS} \times \bar{h}}{|\bar{1}R_{CS} \times \bar{h}|}$$

$$\bar{1}R_{ST} = \bar{1}h \times \bar{1}C$$

where

$\bar{R}_{INJ}$ = position vector at injection into orbit

$\bar{V}_{INJ}$ = velocity vector at injection into orbit

$\bar{1}R_{ST}$ = intersection of trajectory plane and the plane perpendicular to the trajectory through the center of the shadow cone.

The orbital elements of the conic are computed on the basis of the position and velocity at injection into orbit, $\bar{R}_{INJ}$, $\bar{V}_{INJ}$.

The vehicle's position and velocity at the intersection of the trajectory plane and the plane perpendicular to the trajectory through the center of the shadow ($\bar{R}_{ST}$, $\bar{V}_{ST}$) are computed with the Two-Body Program (Reference 2), Δ true anomaly option.

NOTE: $\bar{R}_{ST}$ is the vehicle's closest approach to the center of the shadow cone.

The Δ true anomaly, $\Delta\theta$ (Figure 2-6), is formed simply by computing the angle between $\bar{1}R_{INJ}$ and $\bar{1}R_{ST}$.

The radius of the shadow cone, B, and the vehicle's distance from the center of the shadow cone at its closest point, R_C, are formed as follows:

$$B = R_S - (\bar{R}_{ST} \cdot \bar{1}R_{CS}) \tan \alpha$$

$$R_C = |\bar{R}_{ST} \times \bar{1}R_{CS}|$$

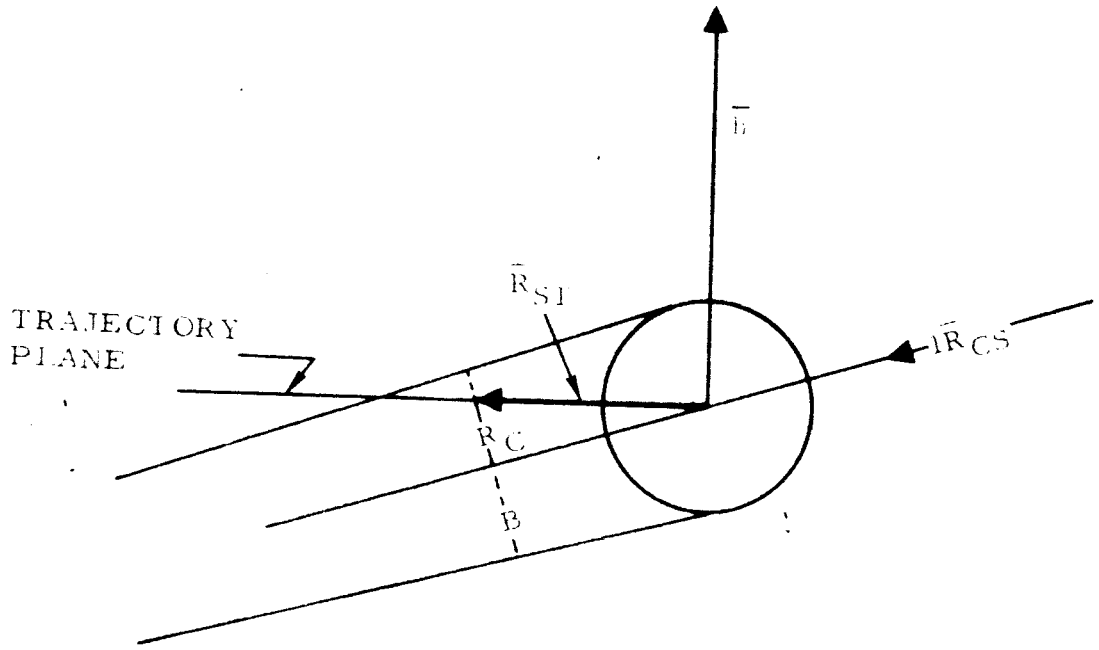

Figure 2-5. Intersection of Trajectory Plane

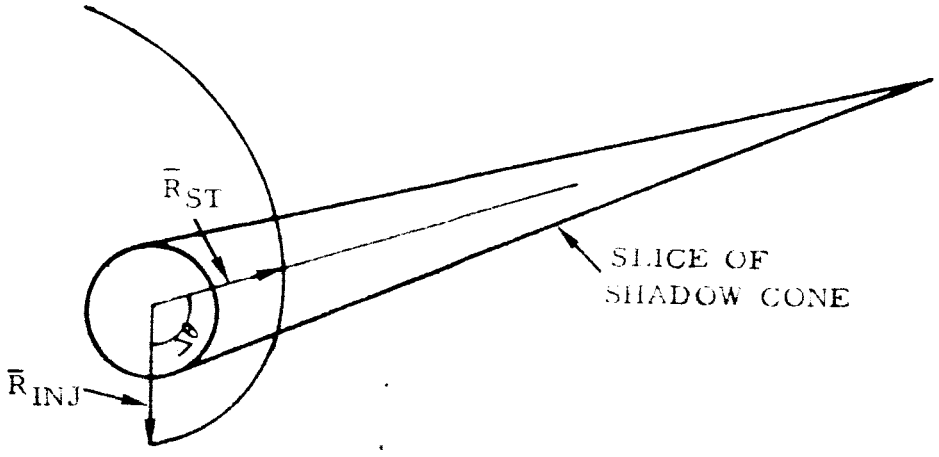

Figure 2-6. Closest Approach to Shadow Cone

where

R_S = radius of central body at edge of shadow

$\bar{R}_{ST}$ = the position of the vehicle's closest approach to the center of the shadow cone

When $R_C > B$, the vehicle does not pass through the shadow cone. When $R_C < B$, the shadow entry and exit points are approximated by computing the chord of the circle formed by slicing the shadow cone at $\bar{R}_{ST}$ (Figure 2-7). Computations are as follows.

$$D = B \sin \left[\cos^{-1} (R_C / B) \right]$$

where

D = one half the chord of the circle

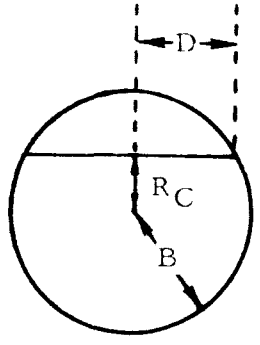

Figure 2-7. Chord of Circle by Slicing Shadow Cone

The approximation of the entry and exit positions is completed by computing the angles β_1 and β_2 between $\bar{R}_{ST}$ and the edge of the shadow as defined by D (Figure 2-8). An adjustment is also made to compensate for the error in the model caused by the flight path angle at $\bar{R}_{ST}$.

β_1 and β_2 are computed as follows.

$$\beta_1 = -\tan^{-1} \left(\frac{D + A_D}{|\bar{R}_{ST}|} \right)$$

$$\beta_2 = \tan^{-1} \left(\frac{D - A_D}{|\bar{R}_{ST}|} \right)$$

where

$$A_D = |D - D \cos \gamma_S| \text{ and takes the sign of } \gamma_S$$

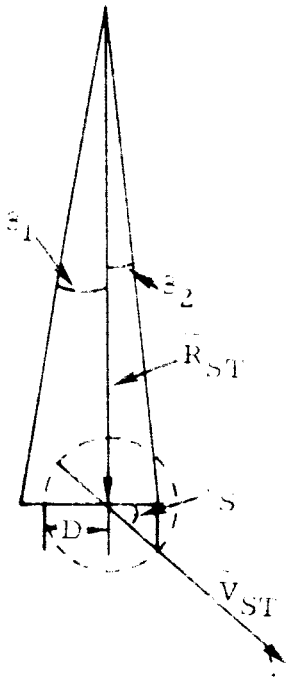

Figure 2-8. Approximation of Entry and Exit Positions

The final computation of the shadow entry and exit points is made by adjusting β_1 and β_2 through an iteration until the exact intersection is found. β is computed as follows.

a. Compute $\bar{R}_T$, $\bar{V}_T$, and T at β degrees from $\bar{R}_{ST}$.

NOTE: β is initialized as either β_1 or β_2.

b. Compute radius of shadow cone B and the vehicle's distance at $\bar{R}_T$ from the center of the shadow cone.

$$B = R_S - (\bar{R}_T \cdot \bar{1}R_{CS}) \tan \alpha$$

$$R_C = |\bar{R}_T \times \bar{1}R_{CS}|$$

c. Compute distance from $\bar{R}_T$ to edge of shadow, M_S.

$$M_S = B - R_C$$

d. Compute corrected β.

$$\Delta\beta = \frac{\Delta\beta(M_S)}{R_C - R_{C_{n-1}}}$$

$$\beta = \beta + \Delta\beta$$

$$R_{C_{n-1}} = R_C$$

Computations a through d are repeated until M_S goes to zero.

SECTION 3

INPUT/OUTPUT

The Lunar Targeting Program II and the N-Body Space Program have been modified to compute a space vehicle's entry into, and exit from, the shadow cones caused by interfering bodies coming between the vehicle and the sun. The use of these programs with the shadow option requires no additional input to either program.

Input to the Lunar Targeting Program II and the N-Body Space Program is explained in detail in References 1 and 3.

Examples of shadow output follow. Pages 3-2 and 3-3, output from the N-Body Program, show entry into, and exit from, the earth's shadow while in an eccentric earth orbit. Pages 3-4 and 3-5, output from the Lunar Targeting Program II, show shadow entry and exit during the geocentric phase of the trajectory.

```
      DTK=100.,
      TC=0.,
      IATSTP=0,
      KPRNT=2,
      ICOSW=4,
      IPERE=1,
      PERTUR=$20,
      C2=.0032469,
      C3=.575E-5,
      C4=.9E-5,
      NSEKT=1,
      DATE=1964.,12.,8.,.62416810,
      UDATE=1964.,12.,8.,.62416810,
      ICENT=4,
      IPERE=1,
      DTK=200.,
       R= - 4189.0890, 7755.6768 , -2822.9996,
       V = - 6.8167842, .18429465 , 1.8718851,
            REQAT =6378.165,
       TOLRE=127.,
        GMPLAN= .13271544E12,,, 398603.20,,  4906.7589,, .12671760E9,
        STAGER= 14400.,
        TITAB= 4.,
          *
```

KCYL = 0 YEAR 1964 MONTH 12 DAY 8 HOUR 14.980034

R(1)	R(2)	R(3)	RMAG	V(1)	V(2)	V(3)	VMAG
0.43658344+04	0.76552876+04	-0.28291988+04	0.92557174+04	-0.34437728+01	0.58842714+01	0.18767630+01	0.70715246+01

R2B(1)	R2B(2)	R2B(3)	R2BMAG	V2B(1)	V2B(2)	V2B(3)	V2BMAG
0.43658344+04	0.76552876+04	-0.28291988+04	0.92557174+04	-0.34437728+01	0.58842714+01	0.18767630+01	0.70715246+01

DT	TFF	A(S.MAJ.AXIS)	P(LAT.RECT)	E(ECCEN)	ITERATIONS
0.	0.	0.11034066+05	0.92167439+04	0.40583376-00	0

EXIT SHADOW

POSITION AND VELOCITY IN KM-KM/SEC (EQUAT. COOR. 1950.0) TIME FROM INJ. IN HOURS

R= -0.18194233+04 0.12631016+05 0.70901406+03 V= -0.42937540+01 0.11991010+01 0.25233767+01 T= 0.41228084-00

ENTER SHADOW

POSITION AND VELOCITY IN KM-KM/SEC (EQUAT. COOR. 1950.0) TIME FROM INJ. IN HOURS

R= 0.44037392+04 0.75860794+04 -0.28312953+04 V= -0.34200893+01 0.59285408+01 0.18765534+01 T= 0.31980048+01

EXIT SHADOW

POSITION AND VELOCITY IN KM-KM/SEC (EQUAT. COOR. 1950.0) TIME FROM INJ. IN HOURS

R= -0.19053073+04 0.12644249+05 0.79164965+03 V= -0.42894734+01 0.11464658+01 0.25257715+01 T= 0.36186642+01

Example 1

YEAR 1964 MONTH 12 DAY 5 HOUR 18.980034

KCYL = 72							
R(1)	R(2)	R(3)	RMAG	V(1)	V(2)	V(3)	VMAG
-0.72425359+04	0.12280237+05	0.39829089+04	0.14802773+05	-0.33687661+01	-0.14751144+01	0.20479710+01	0.42093625+01
R2B(1)	R2B(2)	R2B(3)	R2BMAG	V2B(1)	V2B(2)	V2B(3)	V2BMAG
-0.71991082+04	0.12322391+05	0.39226793+04	0.14800537+05	-0.33787563+01	-0.14471888+01	0.20562629+01	0.42117177+01
DT	TFF	A(S.MAJ.AXIS)	P(LAT.RECT)	E(ECCEN)	ITERATIONS		
0.20000000+03	0.14399999+05	0.11030496+05	0.92172796+04	0.40544068-09	1		

Example 1 (continued)

```
ID=SHCHECK1,
RCRSS=21890C.,
GME=398603.2C,
GML=4868.3239,
CUL=5378.165,
GMCU=3986[illegible]3.2C,
CCCT=107.C6612,
CCS=[illegible]4CC.,
CMEGAE=[illegible].98565,
RMCCN=1738.0575,
RCA=6544.7401,
RMAG=6587.5761,
DTI=C.,
DT2=67C.5392.
TCL=.CC2,.CC2,.CC2,
DI=.CCC4,.CCC1,.CCC1,
ATLNCH=35.6,
  BLATLS= 23.5,
ELCNLS=279.4618,
TSLAT=C.,
TSLCN=C.,
KEARTS=1,
KREF=2,
ITRANS=3,
TGTLAT=C.,
TGTLCN=C.,
IFLAG=1,
TAB=1[illegible]4.,4.,3[illegible].,1[illegible].P4511C,
DEL=19=4.,4.,27.,1F.24,
TEE=44.,
IMPACT=1,
     ARC2=27.254454.
     RRC  =6544.7401,
     ,AMMAE=2..
KASNT=2,
       APNTS =1   ,
  NCCE=C,
  AZIMIN=9C.,
 *

 IDENTIFICATION - -  CHECK1
```

Example 2

EARTH SHADOW ON GEOCENTRIC CONIC

POSITION AND VELOCITY AT ENTRY TO SHADOW IN KM-KM/SEC (ECLIPTIC COORDINATES OF 1960.0)
TIME FROM INJECTION INTO ORBIT IN HOURS
R= -0.16086423+05 -0.46455050+04 0.13325205+04 V= -0.39093301+01 -0.56043886+01 0.35304630-00 T= 0.62081192+00

POSITION AND VELOCITY AT EXIT FROM SHADOW IN KM-KM/SEC(ECLIPTIC COORDINATES OF 1960.0)
TIME FROM INJECTION INTO ORBIT IN HOURS
R= -0.27591940+05 -0.28541645+05 0.24158893+04 V= -0.14839908+01 -0.41443004+01 0.14718366-00 T= 0.20129919+01

TOTAL TIME IN SHADOW= 0.13921800+01

MOON SHADOW ON SELENOCENTRIC CONIC

SHADOW NOT ENTERED

GEOCENTRIC GROUP

TL=1964. 4. 27. 21.7541 TI=1964. 4. 27. 23.2792 AZIL = 0.90000003+02

V = 0.10971385+05 AZI = 0.8849785C+02 PTH = 0.20000002+01 LATL = 0.23500000+02 LONL = 0.27946180+03 RMAG = 0.65875756+04

LATS = 0.23459999+02 LONS = 0.27946175+03 LATE = 0.20048928+02 LONE = 0.22639049+03 LATI = 0.23454659+02 LONI = 0.25275331+03

ARC1 = 0. ARC2 = 0.27254454+02 DT1 = 0. DT2 = 0.67053919+03 GME = 0.3986C319+06 GMM = 0.48983238+04

C3 =-0.64129711+00 SMA = 0.62155776+06 ECC = 0.98941438+00 INC = 0.23500001+02 LNGA = 0.16652276+03 PARG = 0.82520995+02

RCA = 0.65795092+04 SLR = 0.13C8937C+05 TA = 0.40214067+01 EA = 0.51218411-02 DAA =-0.22013231+02 RAA = 0.26001471+03

GMA = 0.1822113C+03

SELENOCENTRIC GROUP

TI=1964. 4. 30. 10.8456 TT= 61.0515

θ = 0.27518859+04 AZI = 0.81373123+02 PTH =-0.63818859+02 LAT = 0.88554741-01 LON = 0.10225825-00 RMAG = 0.17405041+04

C3 = 0.19442495+07 SMA =-0.25193904+04 ECC = 0.11669943+01 EINC = 0.86273296+01 LNGA = 0.35951862+03 PARG = 0.11467255+03

RCA = 0.42072397+03 SLR = 0.91170647+03 TA =-0.11408188+03 EA =-0.91523030+00 DAI =-0.48485468+01 RAI = 0.32552322+03

BT = 0.15070054+04 BR =-0.16086960+03 B = 0.86439177+03 0.12306448+04 0.18781320+03

XM =-0.65195757+05 YM =-0.37029947+06 ZM =-0.15201282+06 DXM = 0.95781345+00 DYM =-0.10716034-00 DZM =-0.13870366-00

DRM =-0.22013231+02 RAM = 0.26001471+03 DRE = 0.14303510-02 RAE = 0.40067775+02 FL =-0.92583749+01 DT = 0.23792408+03

Example 2 (continued)

SECTION 4

PROGRAM SUBROUTINES

Section 4 describes the subroutines of the Shadow Constraint Program.

AIMP Subroutine - Figure 4-1

PURPOSE

AIMP subroutine computes two-body orbits with delta true anomaly ($\Delta\theta$) as the independent variable.

COMPUTATIONAL SEQUENCE - See Two-Body Program, Reference 2.

INPUT

KORB - Orbit classification flag

DELTV - Delta true anomaly

ROVEC - Input position vector

VOVEC - Input velocity vector

GM - Mass of body times gravitational constant

AMAJR - Semimajor axis

ECCN2 - Eccentricity squared

PLATR - Semilatus rectum

OUTPUT

RIVEC - Output position vector

VIVEC - Output velocity vector

NREVS - Number of revolutions in orbit

IMPCT - Error out flag

TIMEI - Time in orbit from $\bar{R}_o$ to $\bar{R}_I$.

SUBROUTINES USED

CPDELV, RVRDV, RIDEV, RCOMP, FGDEV, TRPAR, RIVI, TRELH

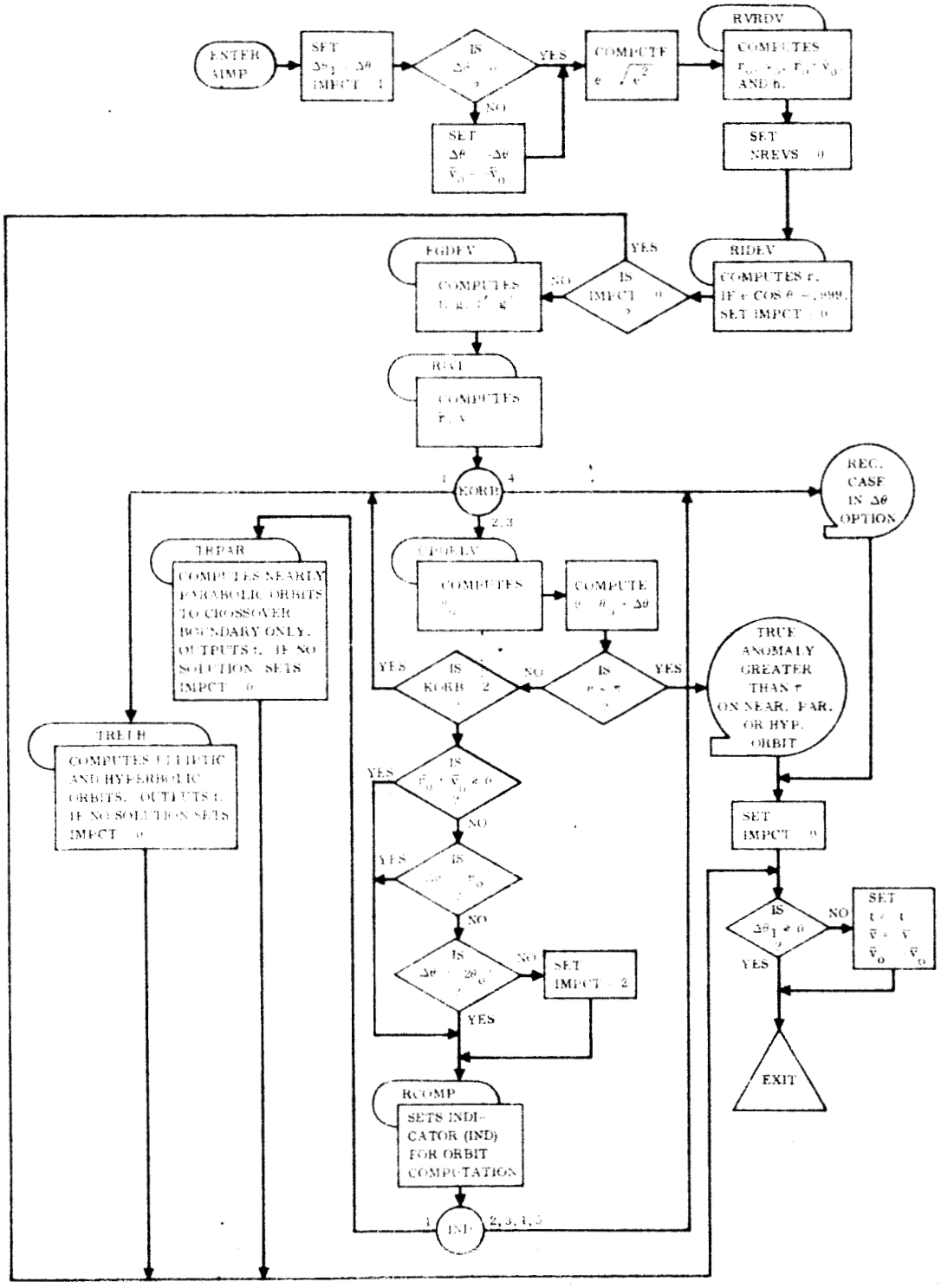

Figure 4-1. AIMP Subroutine

AZGAM Subroutine - Figure 4-2

PURPOSE

AZGAM subroutine computes flight path angle and azimuth for a given unit position and unit velocity.

COMPUTATIONAL SEQUENCE

a. Compute flight path angle

$$\gamma = \sin^{-1}(\bar{1}R \cdot \bar{1}V)$$

b. Compute azimuth

$$\bar{E} = \bar{1}Z \times \bar{1}R$$

$$\bar{1}V = \bar{1}R \times \bar{1}E$$

$$\bar{V}_P = (\bar{1}V \cdot \bar{1}E)\,\bar{1}E + (\bar{1}V \cdot \bar{1}V)\,\bar{1}V$$

$$\cos(AZ) = \bar{1}V_P \cdot \bar{1}E$$

$$\sin(AZ) = |\bar{1}V_P \times \bar{1}E|$$

INPUT

ONER - Unit position vector

ONEV - Unit velocity vector

OUTPUT

AZMUTH - Azimuth

GAMMA - Flight path angle

SUBROUTINES USED - None

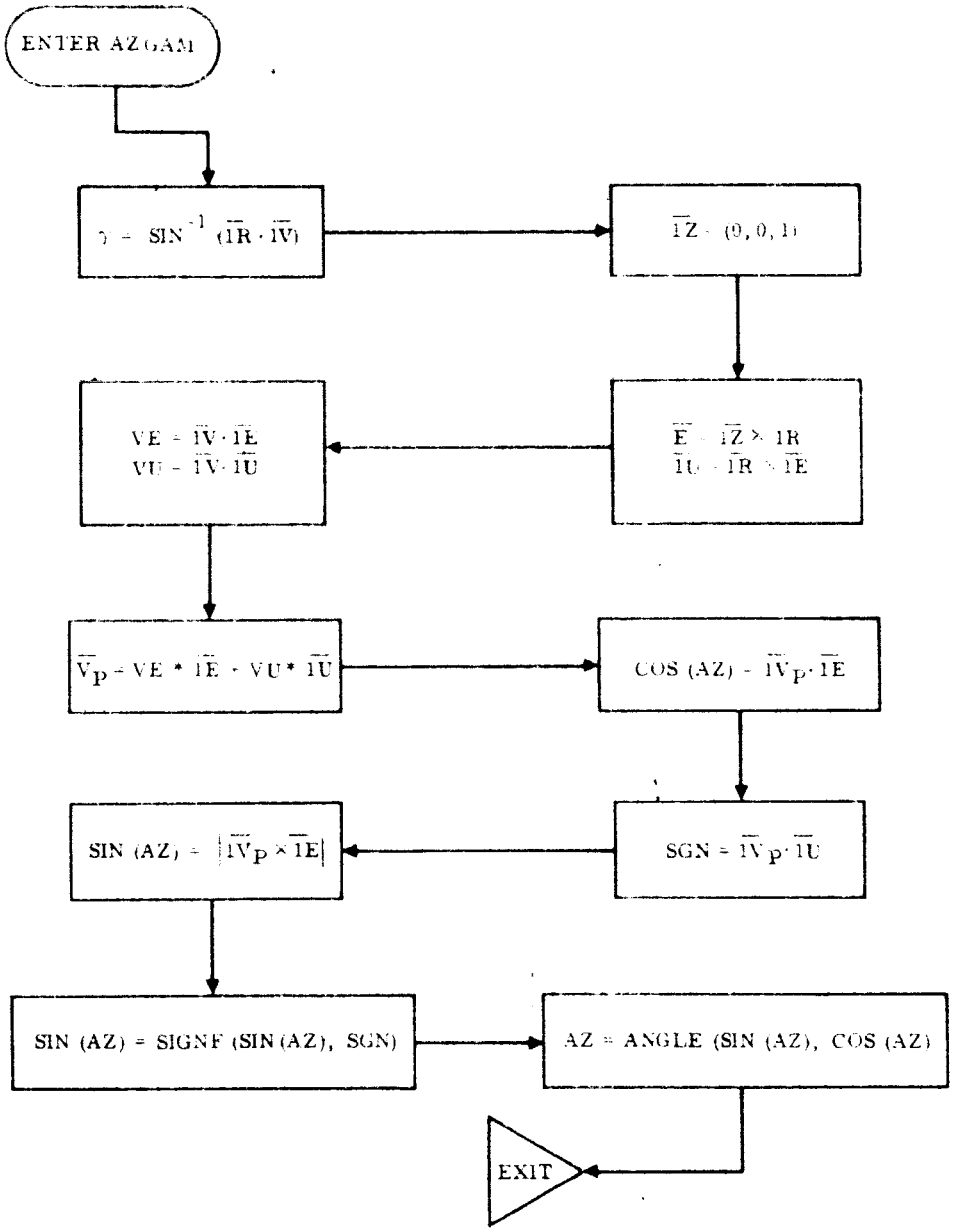

Figure 4-2. AZGAM Subroutine

DELG Subroutine - Figure 4-3

PURPOSE

DELG subroutine computes the total acceleration due to gravity and its time derivative (oblate model).

COMPUTATIONAL SEQUENCE

a. Compute acceleration due to gravity

$$\bar{G} = \frac{-GM}{R^3}\left[(1-Q)\,\bar{R} - \Delta P\,(\bar{1}\omega_{ei})\right]$$

b. Compute time derivative of acceleration due to gravity

$$\dot{\bar{G}} = \frac{GM}{R^3}\left[(Q-1)\,\bar{V} - \frac{3(\bar{R}\cdot\bar{V})\,R(\bar{G})}{GM} + \bar{R}(\dot{Q}) + \Delta\dot{P}(\bar{1}\omega_{ei})\right]$$

INPUT

R	-	Position vector
V	-	Velocity vector
GM	-	Mass of central body times gravitational constant
WEI	-	Body's rotational rate
C2	-	Oblate constant
C3	-	Not used
C4	-	Not used
RO	-	Position - Injection into orbit

OUTPUT

G	-	Acceleration due to gravity
GD	-	Time derivative of acceleration due to gravity

SUBROUTINES USED - None

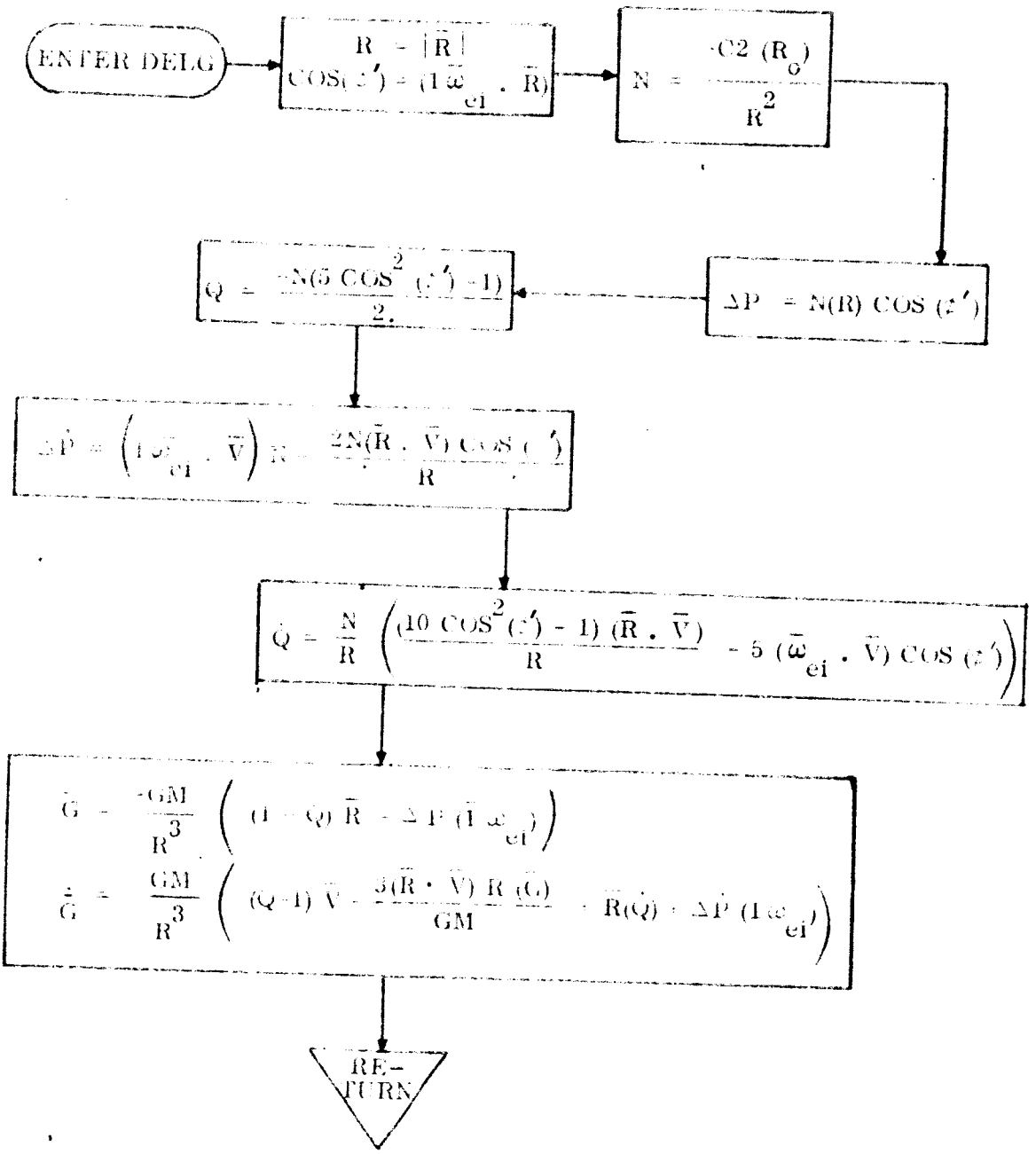

Figure 4-3. DELG Subroutine

EGEN Subroutine - Figure 4-4

PURPOSE

EGEN subroutine computes the ephemeris of a specified planet; Mercury through Saturn, plus the moon.

COMPUTATIONAL SEQUENCE

a. Compute the orbital elements of the body whose orbit is being computed.

b. Compute the position and velocity of the specified body as a function of time.

c. Make necessary coordinate transformations and unit conversions to obtain output in desired system (refer to Ephemeris Generator Program, Reference 4).

INPUT

KBODY = Number of body for which output is desired

DAJUL = Time of desired output, input in days since epoch 1950.0

KPRNT = Debug print flag

OUTPUT

R = Vector position of body

V = Vector velocity of body

SUBROUTINES USED

BMRC, BVNUS, BSUN, BMRS, BJPT, BSTR, EG9IN, RVCMP, DTTEQ

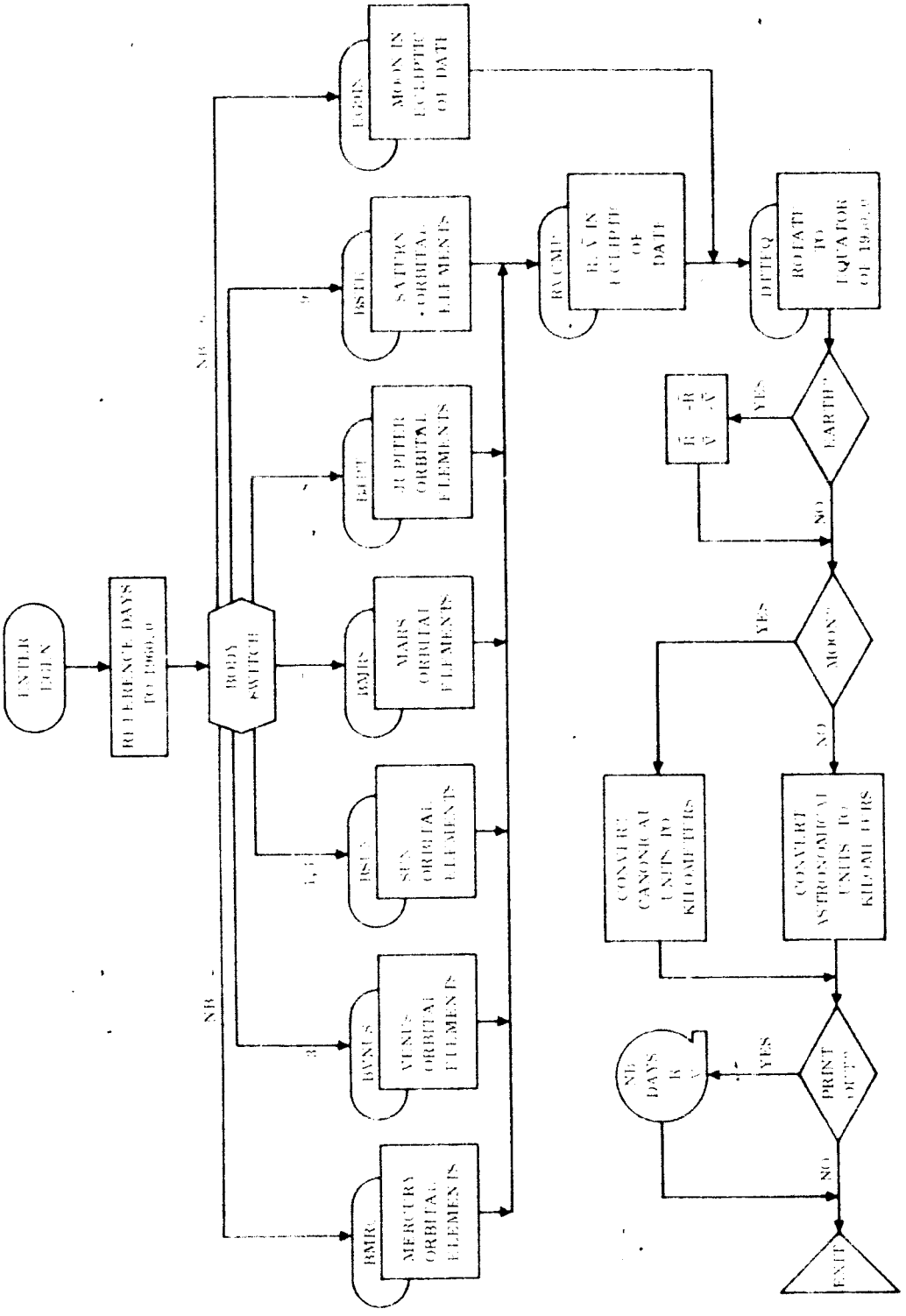

Figure 4-4. EGEN Subroutine

ITSHAD Subroutine - Figure 4-5

PURPOSE

ITSHAD subroutine refines the computed intersection of the trajectory conic and the shadow cone by forcing, through iteration, the distance of the vehicle from the center of the shadow cone to compare with the radius of the cone.

COMPUTATION SEQUENCE

a. Compute orbital elements of trajectory.

b. Compute intersection of trajectory and shadow cone, based on a specified angle (β) from the trajectory's closest approach to the center of the cone.

c. Compare radius of cone with the vehicle's distance from center of cone to form miss.

d. Correct β and repeat steps b through d until miss goes to zero.

INPUT

RO = Position - injection into orbit

VO = Velocity - injection into orbit

GM = Mass of planet times gravitational constant

BETA = Angle from injection to center of shadow

RS = Radius of planet

URSUN = Unit vector position of planet relative to sun

TANALF = Tangent of shadow cone angle

KPRNT = Debug print flag (four or greater for print)

OUTPUT

R = Position at edge of shadow

V = Velocity at edge of shadow

TIME = Time at edge of shadow

NOSOL = No solution flag; zero is no solution

SUBROUTINES USED

ORBEL, AIMP

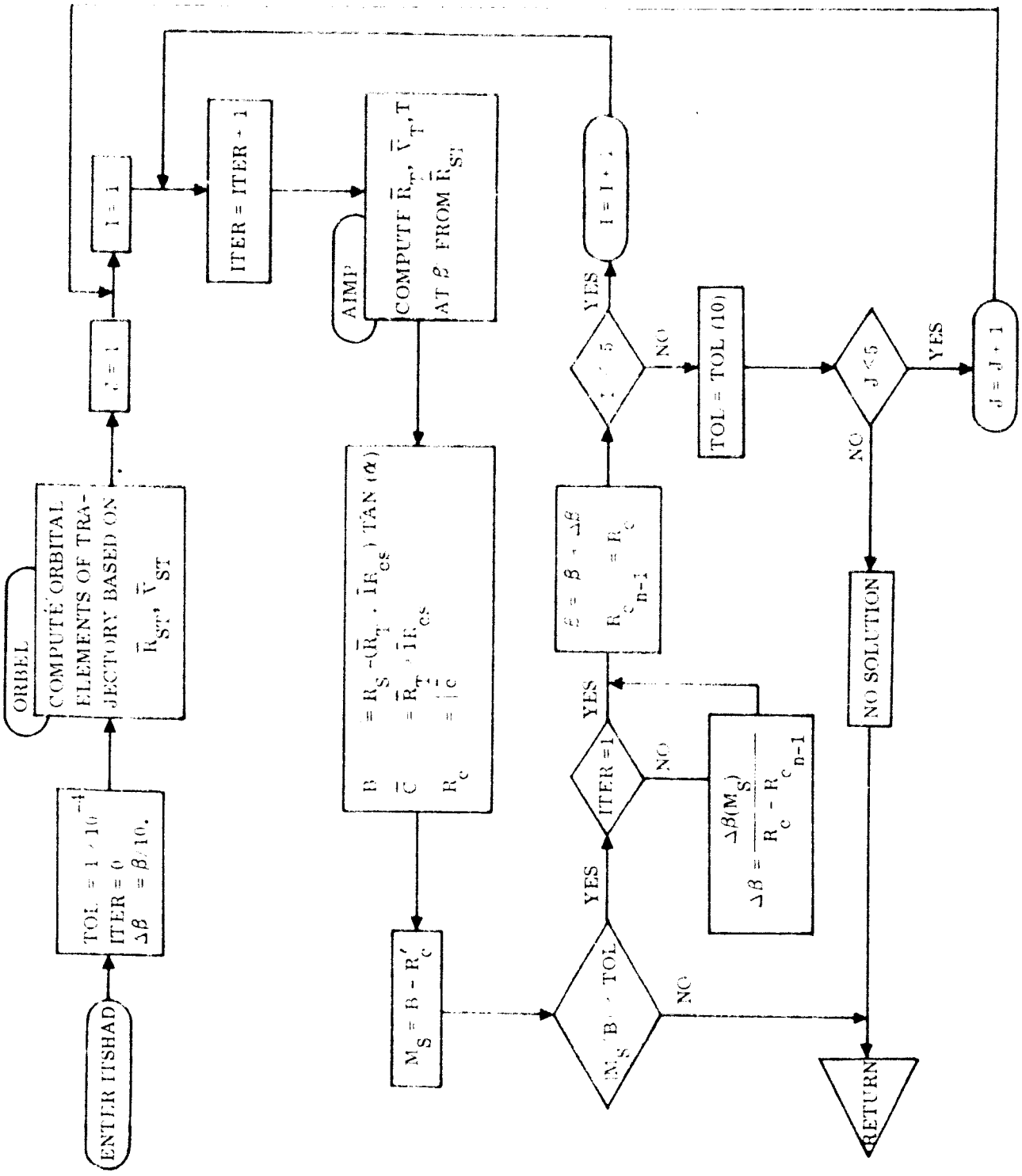

Figure 4-5. ITSHAD Subroutine

INTRP Subroutine - Figure 4-6

PURPOSE

INTRP subroutine interpolates for shadow entry and exit positions.

COMPUTATIONAL SEQUENCE

a. Compute radius of shadow cone, B.

b. Form the miss by comparing the vehicle's distance from the center of the shadow cone with the radius of the cone SMISS = R_C - B.

c. Compute a correction in time, based on the computed miss.

d. Interpolate for vehicle position and velocity, using the corrected time.

e. Recompute the vehicle's distance from the center of the shadow cone and repeat steps a through e until the miss goes to zero.

INPUT

RK	- Position, N
VK	- Velocity, N
RM	- Position, N - 1
VM	- Velocity, N - 1
DT	- Time step
TFF	- Time at position N
RSINTH	- R SIN(THETA), distance from center of shadow cone
RSINTH1	- R SIN(THETA) N - 1
URSUN	- Unit sun to central body vector
TANALF	- Tangent of shadow cone angle
RS	- Radius of ellipsoid (earth)
KPRNT	- Three or greater for debug print

OUTPUT

T	- Time of free fall at edge of shadow
RT	- Position at edge of shadow
VT	- Velocity at edge of shadow

SUBROUTINES USED

DELG

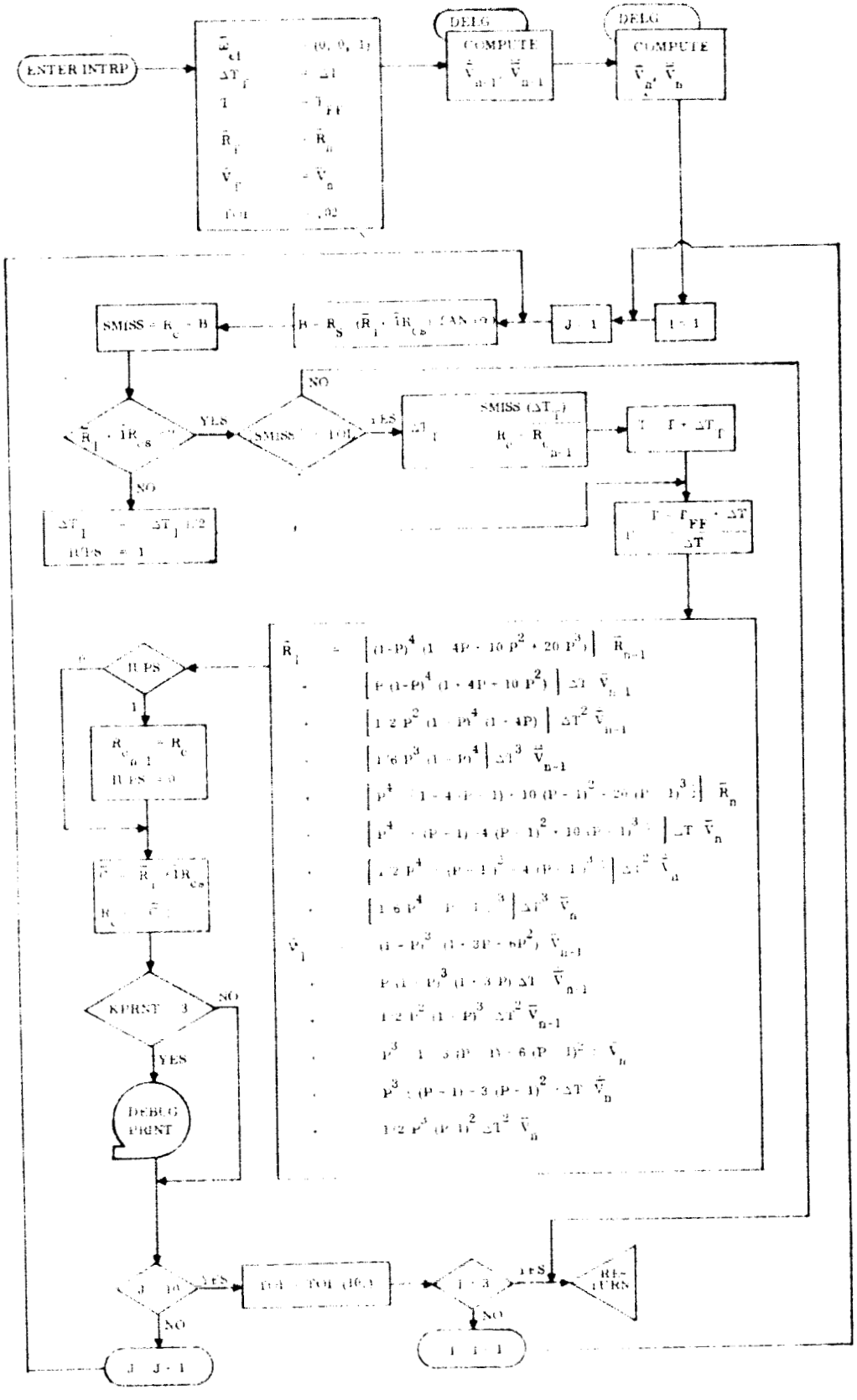

Figure 4-6. INTRP Subroutine

LATLON Subroutine - Figure 4-7

PURPOSE

LATLON subroutine computes latitude and longitude for a given unit position vector.

COMPUTATIONAL SEQUENCE

a. Compute latitude

$$B = \tan^{-1}\left[\bar{1}R(3)/\sqrt{\bar{1}R(1)^2 + \bar{1}R(2)^2}\right]$$

b. Compute longitude

$$\sin(L) = |\bar{1}X \times \bar{1}R_p|$$

$$\cos(L) = \bar{1}R_p \cdot \bar{1}X$$

INPUT

ONER - Unit position vector

OUTPUT

FLONG - Longitude

BLAT - Latitude

SUBROUTINES USED - None

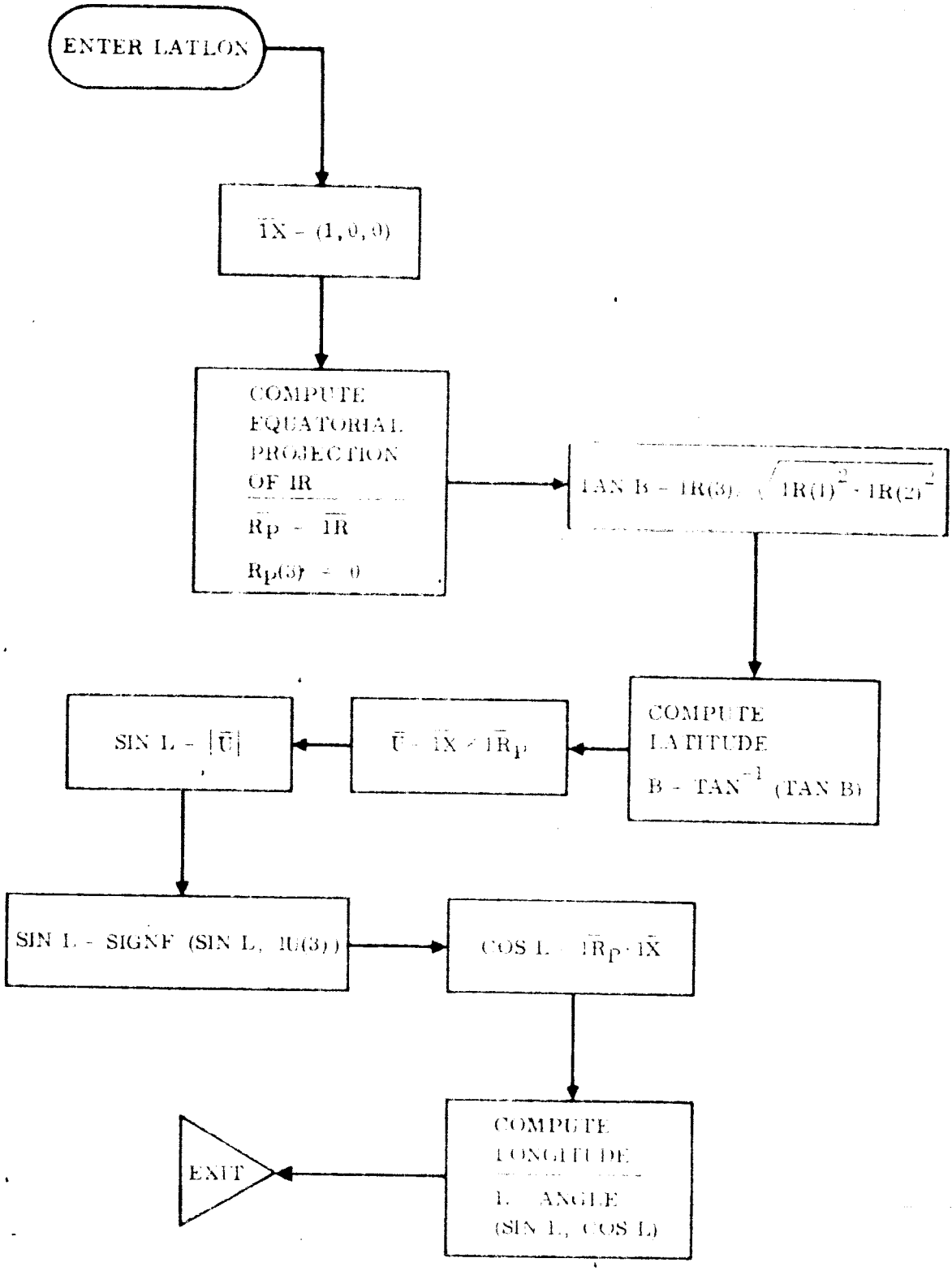

Figure 4-7. LATLON Subroutine

OBR Subroutine - Figure 4-8

PURPOSE

OBR subroutine computes radius of ellipsoid based on semimajor and semiminor axes and input latitude.

COMPUTATIONAL SEQUENCE

Compute radius of ellipsoid

$$R_S = \frac{ab}{\sqrt{b^2 + (a^2 - b^2)\sin^2(\phi)}}$$

INPUT

A - Semimajor axis

B - Semiminor axis

BLAT - Geocentric latitude

OUTPUT

RS - Radius of ellipsoid

SUBROUTINES USED - None

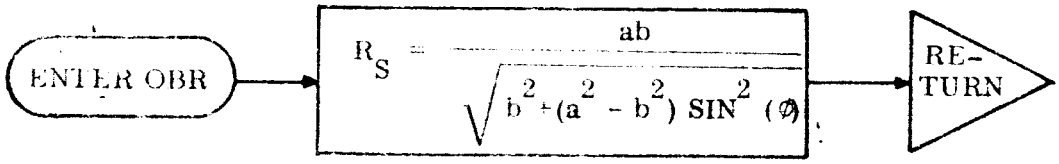

Figure 4-8. OBR Subroutine

ORBEL Subroutine - Figure 4-9

PURPOSE

ORBEL subroutine computes the orbital elements a, p, and e of the two-body trajectory based on an input position and velocity ($\bar{R}_o$ and $\bar{V}_o$).

COMPUTATIONAL SEQUENCE

a. Compute $\bar{R}_o \cdot \bar{V}_o$, h, and p.

b. Compute e.

c. Test for significance loss.

d. Classify orbit (elliptic, hyperbolic, nearly parabolic, or nearly rectilinear).

Refer to Two-Body Program, Reference 2.

INPUT

ROVEC - Input position vector

VOVEC - Input velocity vector

GM - Mass of body times gravitational constant

OUTPUT

AMAJR - Semimajor axis

ECCN2 - Eccentricity squared

PLATR - Semilatus rectum

KORB - Flag specifying classification of orbit

SUBROUTINES USED

RVRDV, DECPT

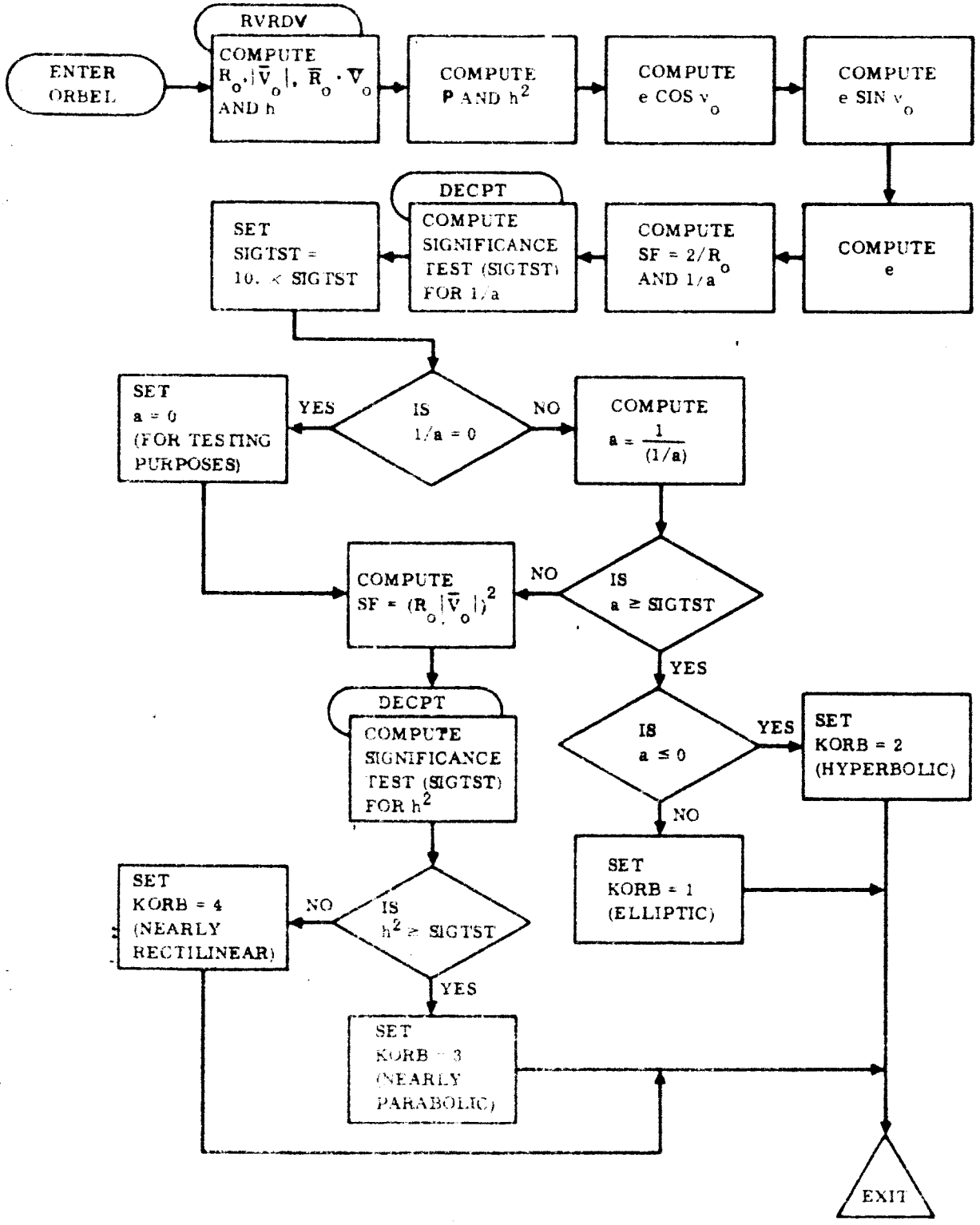

Figure 4-9. ORBEL Subroutine

RT1950 Subroutine - Figure 4-10

PURPOSE

RT1950 subroutine transforms input vector from mean equatorial coordinates of 1950.0 to true equatorial coordinates of date, and vice versa.

COMPUTATIONAL SEQUENCE

a. Compute nutation in longitude ($\Delta\ell$) and nutation in obliquity ($\Delta\epsilon$) and form matrix of rotation representing the transformation from a mean to a true system. Matrix [N].

b. Compute matrix of transformation representing change of epoch from 1950.0 to date. Matrix [A].

c. Transform the input vector by applying matrix [N] and matrix [A]. (Refer to N-Body Space Program, Reference 1.)

INPUT

VEC - Input vector

INVERT - Switch which determines if transformation is from mean equatorial coordinates of 1950.0 to true equatorial coordinates of data or vice versa

DAJUL - Time in Julian days since 1950.0

OUTPUT

RVEL - Output vector

SUBROUTINES USED - None

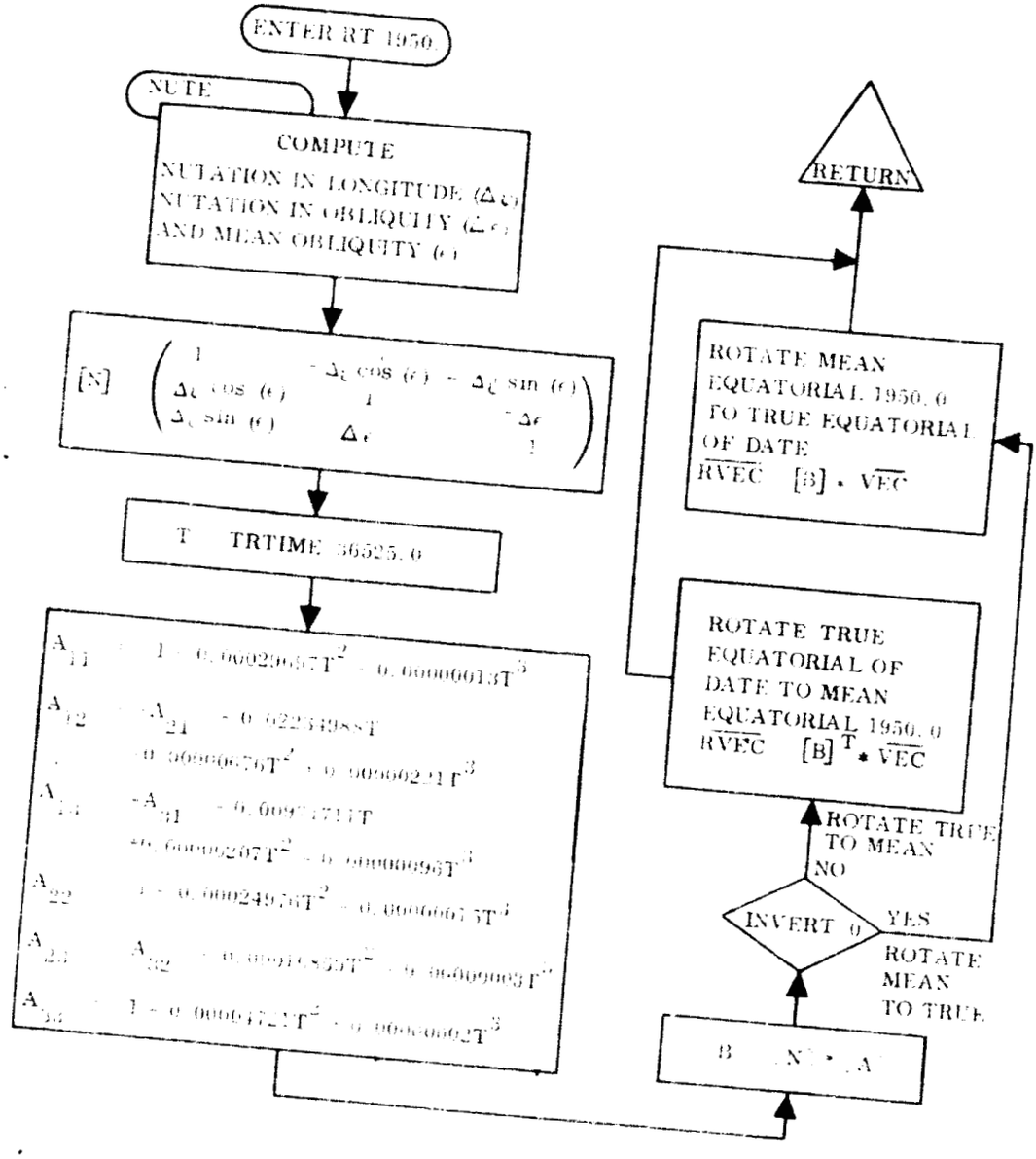

Figure 4-10. RT1950 Subroutine

SHAD1 Subroutine - Figure 4-11

PURPOSE

SHAD1 subroutine computes the positions and times of a space vehicle's entry to, and exit from, the shadow cones caused by interfering bodies (used with N-Body Programs).

COMPUTATIONAL SEQUENCE

a. Compute ephemeris of interfering body relative to the sun.

b. Compute radius of shadow cone

$$B = R_S - (\bar{R} - \bar{1}R_{CS}) \tan \alpha$$

c. Compute the vehicle's distance from the center of the shadow cone

$$R_C = |\bar{R} \times \bar{1}R_{CS}|$$

d. Compare R_C and B to determine if vehicle has crossed edge of shadow.

e. Interpolate for entry or exit points.

INPUT

REM - Vehicle position

VME - Vehicle velocity

TFF - Time of free fall

DT - Time step

DAJUL - Time in days since 1950.0

KPRNT - Three or greater for debug print

KCYCLE - Integration cycle

KCBODY - Central body

OUTPUT

RT - Position at edge of shadow

VT - Velocity at edge of shadow

T - Time of free fall at edge of shadow

ISHAD=0 - Coming into shadow

ISHAD=1 - Going out of shadow

SUBROUTINES USED

EGEN, RT1950, LATLON, OBR, INTRP

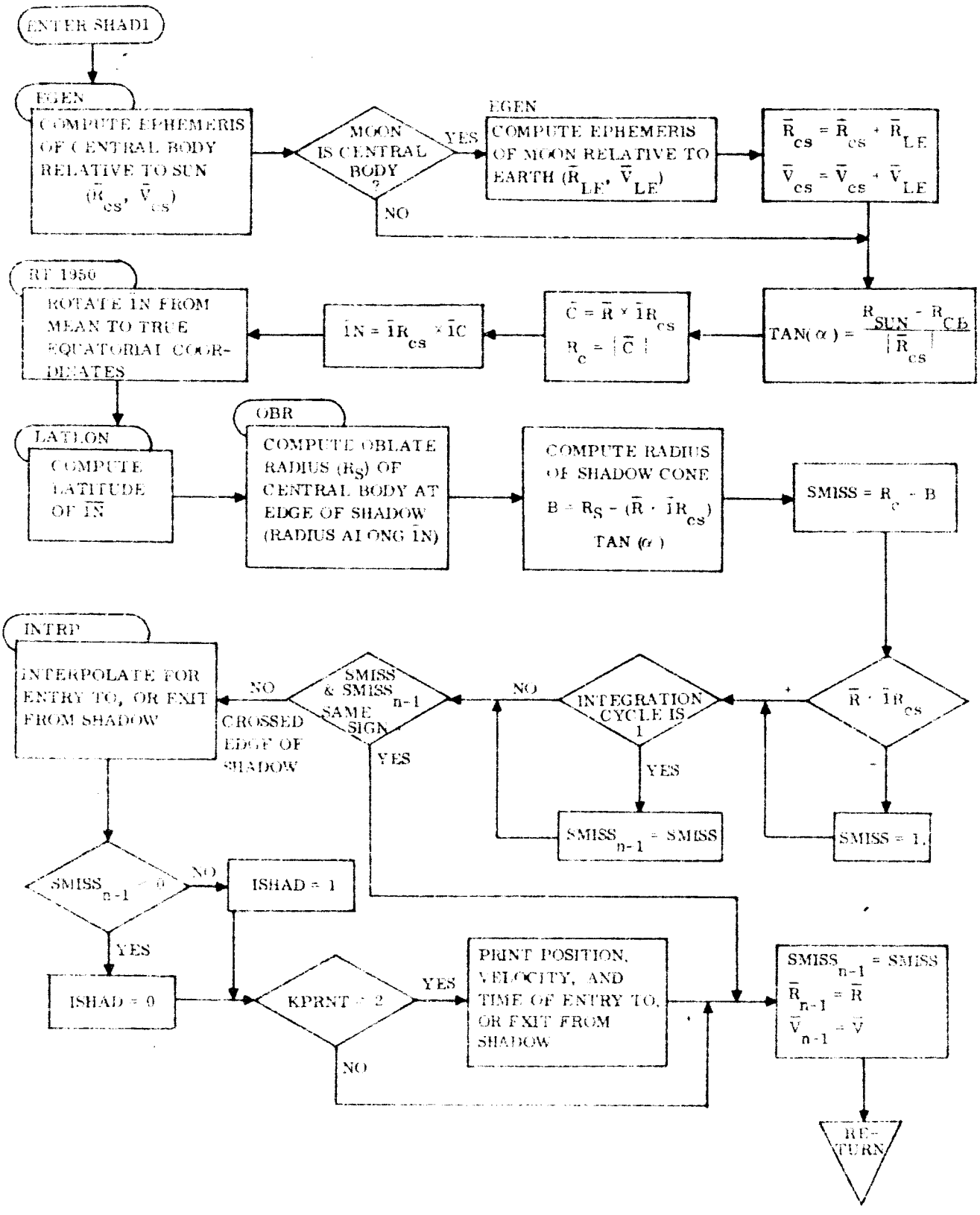

Figure 4-11. SHAD1 Subroutine

SHAD2 Subroutine - Figure 4-12

PURPOSE

SHAD2 subroutine computes the positions and times of a space vehicle's entry to, and exit from, the shadow cone caused by the central body where the trajectory is defined by a conic generated from an $\overline{R}$ and $\overline{V}$ of injection into orbit.

COMPUTATIONAL SEQUENCE

a. Compute the intersection of the shadow cone and the plane perpendicular to the trajectory passing through the center of the shadow cone.
b. Compute the angle from injection to the center of the shadow in the trajectory plane.
c. Form orbital elements of conic trajectory.
d. Compute position of vehicle's closest approach to the center of the shadow cone.
e. Compute radius of shadow cone and the vehicle's distance from the center of the shadow cone at its closest point.
f. Form the chord of the circle formed by slicing the shadow cone at the vehicle's closest approach to the center of the cone.
g. Compute the angles from the vehicle's closest approach to the center of the cone to the edge of the cone as defined by the chord of the slicing circle.
h. Compute exact intersection through an iteration which zeroes out the remaining miss from the edge of the cone.

SHAD2 (continued)

INPUT

RINJ - Injection position

VINJ - Injection velocity

RSUN - Position of planet relative to the sun

GM - Mass of planet times gravitational constant

RS - Radius of planet

KPRNT - Debug print flag (four or greater for print)

TMSHD - Time from injection to center of shadow used to update ephemeris computation. (If TMSHD is zero, there is no shadow.)

IBOD - Body causing shadow (1 = earth, 2 = moon)

OUTPUT

REX - Position exit from shadow

VEX - Velocity exit from shadow

TEX - Time of exit from shadow (in seconds from injection into orbit)

REN - Position entry into shadow

VEN - Velocity entry into shadow

TEN - Time of entry into shadow (in seconds from injection into orbit)

NOSOL - No solution flag, zero is no solution

SUBROUTINES USED

VANGLE; ORBEL, AIMP, AZGAM, ITSHAD

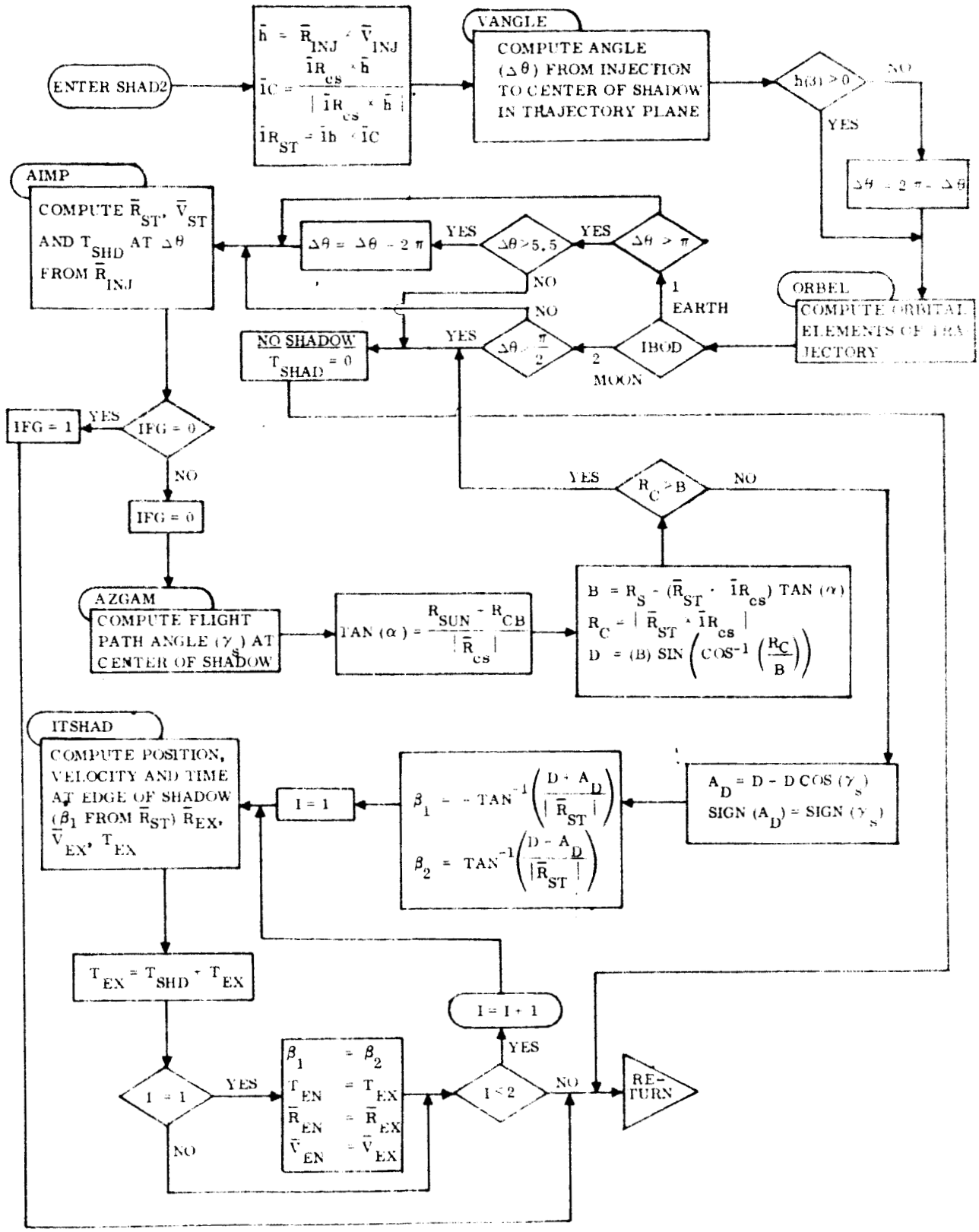

Figure 4-12. SHAD2 Subroutine

VANGLE Subroutine - Figure 4-13

PURPOSE

VANGLE subroutine computes the angle between two vectors.

COMPUTATIONAL SEQUENCE

a. Compute sine and cosine of angle based on vector dot and vector cross.

b. Compute angle from sine and cosine.

INPUT

R1 - Vector 1

R2 - Vector 2

OUTPUT

ANG - Angle

SUBROUTINES USED

ANGLE

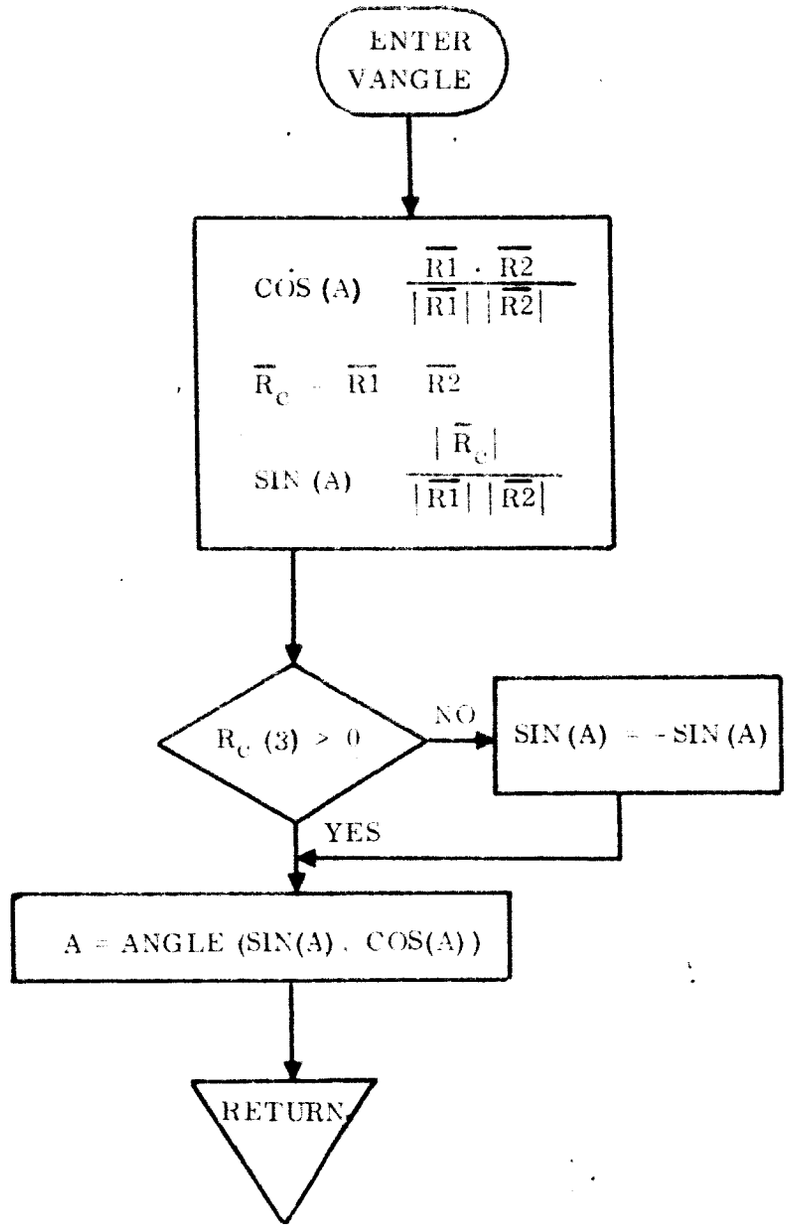

Figure 4-13. VANGLE Subroutine

APPENDIX A

GLOSSARY

a	semimajor axis of earth
b	semiminor axis of earth
B	radius of shadow cone
C_2	oblate gravity constant
GM	mass of central body times gravitational constant
$\bar{G}$	acceleration due to gravity
$\dot{\bar{G}}$	time derivative of acceleration due to gravity
$\bar{h}$	angular momentum vector
$\bar{R}$	vehicle position vector
$\bar{R}_{INJ}$	position - injection into orbit
R_C	vehicle's distance from center of shadow
R_{CB}	radius of the central body
$\bar{R}_{CS}$	vector position of central body relative to the sun
$\bar{R}_{ST}$	intersection of trajectory plane and the plane perpendicular to the trajectory through the center of the shadow cone
$\bar{R}_T$	position at edge of shadow
R_{SUN}	radius of sun
T_{FF}	time of free fall
$\bar{V}$	vehicle velocity vector
$\bar{V}_{CS}$	vector velocity of central body relative to sun
$\bar{V}_{INJ}$	injection velocity
$\bar{V}_T$	velocity at edge of shadow
α	shadow cone angle
β	angle from $\bar{R}_{ST}$ to edge of shadow
ΔT	time step
ϕ_T	latitude
$\bar{\omega}_{EI}$	earth's rotation vector

APPENDIX B

PROGRAM LISTING

```
*       FORTRAN
CAIMP   ROUTINE
C       COMPUTES POSITION AND VELOCITY FOR SPECIFIED ANGLE(DELTV)
C       CALLING SEQUENCE
        SUBROUTINE AIMP(KPRNT,KORB1,DELTV,ROVEC,VOVEC,GMI,AMAJR,ECCN2,
       *PLATR,RIVEC,VIVEC,TIMEI,IMPCT,NREVS)
        DIMENSION ROVEC(3),VOVEC(3),RIVEC(3),VIVEC(3)
      5 NREVS=0
        IMPCT=1
        DELV=DELTV
        IF(DELTV)1,8,8
      1 CONTINUE
        DELTV=-DELTV
        DO 2 J=1,3
      2 VOVEC(J)=-VOVEC(J)
      8 KORB=KORB1
        ECCEN=SQRTF(ECCN2)
        CALL RVRDV(ROVEC,VOVEC,ROMAG,VOMAG,RODVO,HMAG)
        KPRNT1=KPRNT+1
     13 CALL RIDEV(ROMAG,RODVO,GMI,PLATR,DELTV,RIMAG,IMPCT)
        IF(IMPCT)289,289,14
     14 CALL EGDEV(ROMAG,RODVO,GMI,PLATR,RIMAG,DELTV,EFF,GEE,FPRIM,GPRIM)
        CALL RIVI(ROVEC,VOVEC,EFF,GEE,FPRIM,GPRIM,RIVEC,VIVEC)
     17 GO TO(145,200,200,75),KORB
    200 CALL CPDELV(KORB,IMPCT,ROMAG,RODVO,GMI,ECCEN,PLATR,RIMAG,VSUBO,
       *DV)
        VSUBI=VSUBO+DELTV
        IF(3.1415927-ABSF(VSUBI))245,250,250
    245 WOT 6,595
    595 FORMAT(77H TRUE ANOMALY GREATER THAN PI RADIANS ON NEARLY PARABOLI
       *C OR HYPERBOLIC ORBIT)
   1000 IMPCT=0
        GO TO 289
    250 IF(KORB-2)145,145,251
    251 IF(RODVO)256,100,100
    256 IF(DELTV+VSUBO)100,100,257
    257 IF(DELTV+2.*VSUBO)260,100,100
    260 IMPCT=2
    100 CALL RCOMP(IMPCT,ROMAG,RIMAG,ECCN2,PLATR,IND)
        IF(IND-2)50,75,75
     50 CONTINUE
     55 CALL TRPAR(KPRNT,ROMAG,RODVO,GMI,ECCN2,PLATR,RIMAG,EFF,FPRIM,GEE,
       *GPRIM,TIMEI,IMPCT)
        GO TO 289
     75 CONTINUE
        WOT 6,600
    600 FORMAT(36H  RECTILINEAR CASE IN DELTA V OPTION)
        ENERGY=VOMAG**2/2.-GMI/ROMAG
        CALL DEBRE(ENRGY,0,RIMAG,EFF,GEE,FPRIM,GPRIM)
        GO TO 1000
    145 CALL TRELH(KPRNT,ROMAG,RODVO,GMI,AMAJR,RIMAG,KORB,EFF,FPRIM,GEE,
       *GPRIM,ECCEN,NREVS,DELTV,TIMEI,IMPCT)
    289 CONTINUE
        IF(DELV)292,294,294
    292 DELTV=DELV
        DO 293 J=1,3
        VIVEC(J)=-VIVEC(J)
    293 VOVEC(J)=-VOVEC(J)
        TIMEI=-TIMEI
    294 CONTINUE
        IF(KPRNT)15,25,15
     15 GO TO(25,25,20,20,20),KPRNT
     20 CALL PAIMP(DELTV,RIVEC,VIVEC,TIMEI)
     25 RETURN
        END
```

```
*      FORTRAN
CAZGAM ROUTINE
C      COMPUTES GAMMA AND AZIMUTH FOR UNIT POSITION AND VELOCITY VECTORS
C      NOMENCLATURE
C       ONER    - UNIT POSITION VECTOR
C       ONEV    - UNIT VELOCITY VECTOR
C       AZMUTH - AZIMUTH
C       GAMMA   - GAMMA
       SUBROUTINE AZGAM(ONER,ONEV,AZMUTH,GAMMA)
       DIMENSION ONER(3),ONEV(3),ONEE(3),ONEU(3),ONEVP(3),ONEZ(3)
       GAMMA=VDOT(ONER,ONEV,3)
       GAMMA=ASINF(GAMMA)
       ONEZ(1)=0.
       ONEZ(2)=0.
       ONEZ(3)=1.
       CALL VCRS(ONEZ,ONER,ONEE)
       CALL UNIT(ONEE,ONEE,3)
       CALL VCRS(ONER,ONEE,ONEU)
       VE=VDOT(ONEV,ONEE,3)
       VU=VDOT(ONEV,ONEU,3)
       DO 5 I=1,3
     5 ONEVP(I)=VE*ONEE(I)+VU*ONEU(I)
       CALL UNIT(ONEVP,ONEVP,3)
       COSA=VDOT(ONEVP,ONEE,3)
       SGN=VDOT(ONEVP,ONEU,3)
       CALL VCRS(ONEVP,ONEE,ONEVP)
       SINA=MAGF(ONEVP,3)
       SINA=SIGNF(SINA,SGN)
       AZMUTH=ANGLE(SINA,COSA)
       RETURN
       END


*      LIST 8
*      FORTRAN
C   R= INPUT POSITION VECTOR
C WEI= EARTHS ROTATION RATE
C   C2= OBLATE EARTH CONSTANT
C   RO= POSITION-INJECTION INTO ORBIT
C   PHI= GEOCENTRIC LATITUDE
C   V= INPUT VELOCITY VECTOR
C   GM= MASS OF CENTRAL BODY TIMES GRAVITATIONAL CONSTANT
       SUBROUTINE DELG (WEI,R ,V ,GM,C2,C3,C4,RO, G, GD)
       DIMENSION R(3),V(3),UWEI(3),G(3),GD(3)
  5    RMAG=MAGF(R,3)
       R2=RMAG**2
 25    R3=RMAG*R2
       RDOTV=VDOT(R,V,3)
       CALL UNIT(WEI,UWEI,3)
       UWDV=VDOT(UWEI,V,3)
       COSTH=VDOT(UWEI,R,3)/RMAG
       COSTH2=COSTH**2
       RO2=RO*RO
       EN=-C2*RO2/R2
       DP=EN*RMAG*COSTH
       Q=-EN*(5.*COSTH2-1.)/2.
       DPD=UWDV*EN-2.*EN/RMAG*RDOTV*COSTH
       QD=EN/RMAG*((10.*COSTH2-1.)*RDOTV/RMAG-5.*UWDV*COSTH)
       DO 100 I=1,3
       G(I)=-GM/R3*((1.-Q)*R(I)-DP*UWEI(I))
  100  GD(I)=GM/R3*((Q-1.)*V(I)-3.*RDOTV*RMAG/GM*G(I) +R(I)*QD+ DPD*UWEI
      1(I))
  500  RETURN
       END
```

```
*       LIST 8
*       FORTRAN
CEGEN   SOLAR SYSTEM MODEL
        SUBROUTINE   EGEN(NB,DAYS,R,V,KPRNT)
        DIMENSION RR(3),R(3),VR(3),V(3),T(2),DAYS(2)
D       T=DAYS - 3652.
        GO TO(3,1,2,3,86,10,4,5,6),NB
 86     CALL EXIT
 1      CALL BMRC(T,AMN,CE,CA,W,CN,CI,PM)
        GO TO 11
 2      CALL BVNS(T,AMN,CE,CA,W,CN,CI,PM)
        GO TO 11
 3      CALL BSUN(T,AMN,CE,CA,W,CN,CI,PM)
        GO TO 11
 4      CALL BMRS(T,AMN,CE,CA,W,CN,CI,PM)
        GO TO 11
 5      CALL BJPT(T,AMN,CE,CA,W,CN,CI,PM)
        GO TO 11
 6      CALL BSTR(T,AMN,CE,CA,W,CN,CI,PM)
        GO TO 11
 11     CALL RVCMP(AMN,CE,CA,W,CN,CI,PM,RR,VR)
        GO TO 12
 10     CALL EG9IN(T,RR,VR)
 12     D=T+3934.0
        CALL DTTEQ(D,RR,VR,R,V)
        IF(NB-4)33,44,33
 44     DO 45 J=1,3
        R(J)=-R(J)
 45     V(J)=-V(J)
 33     CONTINUE
        IF(NB-6) 88,77,88
 77     DO 78  J=1,3
        R(J)=R(J)/23434.374
 78     V(J)=V(J)/218.83671
 88     DO 89  J=1,3
        R(J)=R(J)*149470734.
 89     V(J)=V(J)*1730.6173
        DAYO=DAYS
        IF(KPRNT-4) 55,66,66
 66     WOT 6,500,DAYO,NB,R,V
 500    FORMAT(14H DAYS,BODY,R,V/E15.8,5X,I2,5X,3E15.8,5X,3E15.8)
 55     CONTINUE
        RETURN
        END
```

```
*       FORTRAN
CITSHAD ROUTINE -ITERATES FOR SHADOW  ENTRY AND EXIT POINTS
C   RO = POSITION -INJECTION INTO ORBIT
C   VO = VELOCITY -INJECTION NTO ORBIT
C   GM = MASS OF PLANET TIMES GRAV. CONST.
C   BETA= ANGLE FROM INJECTION TO CENTER OF SHADOW
C   RS = RADIUS OF PLANET
C   URSUN = UNIT VECTOR POSITION OF PLANET RELATIVE TO SUN
C   TANALF= TANGENT OF SHADOW CONE ANGLE
C   R = POSITION AT EDGE OF SHADOW          (OUTPUT)
C   V = VELOCITY AT EDGE OF SHADOW          (OUTPUT)
C   TIME= TIME AT EDGE OF SHADOW            (OUTPUT)
C   NOSOL= NO SOLUTION FLAG  -ZERO IS NO SOLUTION  (OUTPUT)
C   KPRNT= DEBUG PRINT FLAG (4 OR GREATER FOR PRINT )
       SUBROUTINE ITSHAD( RO,VO,GM,BETA,RS,URSUN,TANALF, R,V,TIME,NOSOL,
      1 KPRNT)
       DIMENSION  RRCRS(3)
       TOL= 1.E-4
       ITER=0
       DBETA = BETA /10.
       CALL ORBEL (RO,VO, GM,A,E,P,KORB)
       DO 45 J=1,5
       DO 25 I=1,5
       ITER=ITER+1
       CALL  AIMP (RO,VO,KORB, BETA,GM, A,E, P, R,V,NREVS, NOSOL, TIME  )
       IF (NOSOL) 7,6,7
     6 WOT 6, 515
   515 FORMAT ( 22H NO SOLUTION FROM AIMP )
       GO TO 125
     7 CONTINUE
       B= RS-VDOT (R,URSUN,3)*TANALF
       CALL VCRS (R, URSUN,RRCRS )
       RSINTH = MAGF(RRCRS,3)
       SMISS= B-RSINTH
       IF (KPRNT-4) 8,522,522
   522 WOT 6, 525, SMISS, BETA
   525 FORMAT ( // 12H SMISS,BETA= 2E15.8)
     8 IF (ABSF(SMISS/B)-TOL) 125,125,9
     9 IF (ITER-1) 13,17,13
    13 DBETA= DBETA /(RSINTH-RSINT1) *  SMISS
    17 BETA= BETA + DBETA
       RSINT1= RSINTH
    25 CONTINUE
       TOL= TOL*10.
    45 CONTINUE
       NOSOL=1
       WOT 6, 535
   535 FORMAT ( 34H NO SOLUTION FROM SHADOW ITERATION  )
   125 CONTINUE
       RETURN
       END
```

```
#     LIST 8
#     FORTRAN
CINTRP  ROUTINE - INTERPOLATES FOR SHADOW ENTRY AND EXIT POINTS
C  RK= POSITION (N)  (INPUT)
C  VK= VELOCITY (N)  (INPUT)
C  RM= POSITION (N-1)   (INPUT)
C  VM= VELOCITY (N-1)   (INPUT)
C  DT= TIME STEP   (INPUT)
C  TFF= TIME AT POSITION N   (INPUT)
C  RSINTH= RSIN(THETA)  (INPUT)
C  RSINT1= RSIN(THETA)  N-1  (INPUT)
C  URSUN=  UNIT SUN TO EARTH VECTOR   (INPUT)
C  T=  TIME OF FREEFALL AT EDGE OF SHADOW  (OUTPUT)
C  RT= POSITION AT EDGE OF SHADOW (OUTPUT)
C  VT= VELOCITY AT EDGE OF SHADOW (OUTPUT)
C  TANALF = TANG. OF SHADOW CONE ANGLE  (INPUT)
C  RS = RADIUS OF ELLIPSOID  (INPUT)
C  KPRNT= 3 OR GREATER FOR DEBUG PRINT  (INPUT)
      SUBROUTINE INTRP ( RK,VK,RM,VM,  DT,  TFF,  RSINTH,RSINT1,URSUN,T,
     1RT, VT, TANALF,RS,KPRNT, C2,GM,RO)
      DIMENSION RK(3),VK(3), RM(3),VM(3), RT(3),VT(3),URSUN(3),RRCRS(3)
      DIMENSION WEI(3),AG(3),AGD(3),AG1(3),AGD1(3)
      WEI(1)=0.
      WEI(2)=0.
      WEI(3)=1.
      TOL= 1.E-2
      DTT=DT
      IUPS=0
      T=TFF
      HK=DT
      HK2=HK*HK
      HK3=HK2*HK
      DO 25 I=1,3
      RT(I)=RK(I)
 25   VT(I)=VK(I)
      CALL DELG (WEI,RM,VM,GM,C2,C3,C4,RO,AG,AGD )
      CALL DELG (WEI,RT,VT,GM,C2,C3,C4,RO,AG1,AGD1)
      DO 65 II=1,3
      DO 55 JJ=1,10
      B= RS- VDOT(RT, URSUN,3)* TANALF
      SMISS= RSINTH-B
      IF (VDOT(RT,URSUN,3)) 28,33,33
   28 DTT= -ABSF(DTT)/2.
      IUPS=1
      GO TO 39
   33 IF (ABSF(SMISS)-TOL) 500,500,35
   35 CONTINUE
      DTT=-SMISS*DTT /(RSINTH-RSINT1)
   39 T=T+DTT
      PJ=(T-TFF+DT)/DT
      PJ2=PJ*PJ
      PJ3=PJ2*PJ
      PJ4=PJ3*PJ
      PJM4=(1.-PJ)**4
      PM12=(PJ-1.)*(PJ-1.)
      PM13=(PJ-1.)*PM12
      PJM3=(1.-PJ)**3
      DO 50 I=1,3
```

CONTINUED

```
      RT(I)=P1M4*(1.+4.*PJ+10.*PJ2+20.*PJ3)*RM(I)+PJ*P1M4*(1.+4.*PJ+10.*
     1PJ2)*HK*VM(I)+PJ2*P1M4*(1.+4.*PJ)/2.*HK2*AG(I)+PJ3*P1M4/6.*HK3*AGD
     2(I)+PJ4*(1.-4.*(PJ-1.)+10.*PM12-20.*PM13)*RK (I)+PJ4*(PJ-1.-4.*PM1
     32+10.*PM13)*HK*VK (I)+PJ4*(PM12-4.*PM13)/2.*HK2*AG1(I)+PJ4*PM13/6.
     4*HK3*AGD1(I)
   50 VT (I)=P1M3*(1.+3.*PJ+6.*PJ2)*VM(I)+PJ*P1M3*(1.+3.*PJ)*HK*AG(I)+P
     1J2/2.*P1M3*HK2*AGD(I)+PJ3*(1.-3.*(PJ-1.)+6.*PM12)*VK (I)+PJ3*(PJ-1
     2.-3.*PM12)*HK*AG1(I)+PJ3*PM12/2.*HK2*AGD1(I)
      IF (IUPS) 88,85,88
   85 RSINT1= RSINTH
   88 IUPS=0
      CALL VCRS (RT,URSUN, RRCRS )
      RSINTH = MAGF (RRCRS,3 )
      IF(KPRNT-3)55,51,51
   51 CONTINUE
      WOT 6, 52,RT,T,SMISS
   52 FORMAT ( 6H RT,T= 4E15.8, 6HSMISS= E15.8 )
   55 CONTINUE
   65 TOL=TOL*10.
      IT=2
  500 RETURN
      END
```

```
*     FORTRAN
CLATLON ROUTINE
C     COMPUTES LATITUDE AND LONGITUDE FOR A GIVEN UNIT POSITION VECTOR
C     NOMENCLATURE
C      ONER    - UNIT POSITION VECTOR
C      BLAT    - LATITUDE
C      FLONG   - LONGITUDE
      SUBROUTINE LATLON(ONER,BLAT,FLONG)
      DIMENSION ONER(3),ONEX(3),ONEU(3),ONERP(3)
      ONEX(1)=1.
      ONEX(2)=0.
      ONEX(3)=0.
      ONERP(1)=ONER(1)
      ONERP(2)=ONER(2)
      ONERP(3)=0.
      TANLT=ONER(3)/SQRTF(ONER(1)**2+ONER(2)**2)
      BLAT=ATANF(TANLT)
      CALL UNIT(ONERP,ONERP,3)
      CALL VCRS(ONEX,ONERP,ONEU)
      SINLON =MAGF(ONEU,3)
      SINLON=SIGNF(SINLON,ONEU(3))
      COSLON=VDOT(ONERP,ONEX,3)
      FLONG=ANGLE(SINLON,COSLON)
      RETURN
      END
```

```
*     FORTRAN
COBR   ROUTINE  -COMPUTES RADIUS OF ELLIPSOID BASED ON SEMIMAJOR AND
C  SEMIMINOR AXES AND INPUT LATITUDE
C  A=SEMIMAJOR AXES  (INPUT)
C  B=SEMIMINOR AXES  (INPUT)
C  BLAT= GEOCENTRIC LATITUDE (INPUT)
C  RS= RADIUS OF ELLIPSOID (OUTPUT)
      SUBROUTINE OBR (A,B, BLAT, RS )
      RS= A*B/SQRTF(B**2+(A**2-B**2)*  SINF(BLAT)**2)
      RETURN
      END
```

```
*       FORTRAN
CORBEL  ROUTINE
C       COMPUTES SEMI-MAJOR AXIS,ECCENTRICITY,SEMI-LATUS RECTUM,
C       AND TYPE OF ORBIT
C       KORB IS INDICATOR FOR TYPE OF ORBIT COMPUTATION.
C       IF KORB=1(ELLIPTICAL),2(HYPERBOLIC),3(PARABOLIC),4(RECTILINEAR)
        SUBROUTINE ORBEL(RABOD,ROVEC,VOVEC,GMI,AMAJR,ECCN2,PLATR,KORB,
       *KPRNT)
        DIMENSION ROVEC(3),VOVEC(3)
        CALL RVRDV(ROVEC,VOVEC,ROMAG,VOMAG,RODVO,HMAG)
        HMAG2=HMAG**2
        PLATR=HMAG2/GMI
        ECSVO=(PLATR-ROMAG)/ROMAG
        ESNVO=HMAG*RODVO/(ROMAG*GMI)
        ECCN2=ECSVO**2+ESNVO**2
        ECCEN=SQRTF(ECCN2)
        SIGFAC=2./ROMAG
        O1DA=SIGFAC-VOMAG**2/GMI
        CALL DECPT(SIGFAC,SIGTST)
        SIGTST=SIGTST*10.
        IF(KPRNT-5)5,19,19
   19   WOT 6,300,O1DA,SIGFAC,SIGTST
  300   FORMAT(14H       1/AMAJR=E15.8,3X,16HSIG. FAC.(2/RO)=E15.8,3X,
       *10HSIG. TEST=E15.8)
    5   IF(O1DA)15,10,15
C       IF O1DA=0,SET AMAJR=0 FOR TESTING TO INDICATE AMAJR IS VERY LARGE
   10   AMAJR=0.0
        GO TO 20
   15   AMAJR=1./O1DA
C       TEST O1DA AGAINST SIGTST TO SEE THAT NO MORE THAN 1 DIGIT WILL BE LOST
C       BY DIFFERENCING. IF NOT,O1DA CAN BE COMPUTED SIGNIFICANTLY.
C       SET KORB=1 OR 2
   50   IF(ABSF(O1DA)-SIGTST)20,52,52
   52   IF(AMAJR)55,55,60
   55   KORB=2
        GO TO 75
   20   SIGFAC=(ROMAG*VOMAG)**2
        CALL DECPT(SIGFAC,SIGTST)
        IF(HMAG2-SIGTST)80,40,40
   80   KORB=4
        GO TO 75
   40   KORB=3
        GO TO 75
   60   KORB=1
   75   CONTINUE
        IF(KPRNT)23,30,23
   23   CALL PRT2B(KPRNT,KORB,ROVEC,VOVEC,GMI,AMAJR,ECCEN,PLATR)
        GO TO(30,30,30,25,25),KPRNT
   25   WOT 6,301,HMAG2,SIGFAC,SIGTST
  301   FORMAT(14H         HMAG2=E15.8,3X,20HSIG. FAC.(RO*VO)**2=E15.8,3X,
       *10HSIG. TEST=E15.8)
   30   CONTINUE
        RETURN
        END
```

```
*     FORTRAN
CRT1950 TRANSFORMATION FROM MEAN 1950 TO TRUE DATE AND VICE VERSA(D.P.)
C     FORTRAN CALLING SEQUENCE
C     CALL RT1950(VN,INVERT,DAJUL,VO)
C NOTE...
C       (1) THE TRANSFORMATION EMPLOYS DOUBLE PRECISION ARITHMETIC
C       (2) LOCATIONS VN AND VO ARE SINGLE PRECISION...CAN BE IDENTICAL
C IF INVERT=0....VN IS VECTOR IN MEAN EQUATOR AND EQUINOX SYSTEM(1950)
C            ....VO IS VECTOR IN TRUE EQUATOR AND EQUINOX SYSTEM(DATE)
C IF INVERT=1....VN IS VECTOR IN TRUE EQUATOR AND EQUINOX SYSTEM(DATE)
C            ....VO IS VECTOR IN MEAN EQUATOR AND EQUINOX SYSTEM(1950)
C DAJUL IS D.P. NO. OF DAYS FROM JAN 1,1950 TO DATE(INTEGER+FRACTION)
      SUBROUTINE RT1950(VN,INVERT,DAJUL,VO)
      DIMENSION VN(3),VO(3),DAJUL(2)
D     DIMENSION EN(3,3),A(3,3),ENA(3,3),T(1),DVN(3),DVO(3)
   10 DO   20    I=1,3
   20 DVN(I)=VN(I)
   30 D=DAJUL+DAJUL(2)
   50 IF (ABSF(D-DSN)-0.1)200,60,60
   60   DSN=D
   70 IMAT=1
   80 CALL NUTE (DAJUL,DEPS,DPSI,EPS)
   90 SINE=SINF(EPS)
  100 COSE=COSF(EPS)
  110 EN(1,1)=1.
      EN(2,1)=+DPSI*COSE
      EN(3,1)=+DPSI*SINE
      EN(1,2)=-DPSI*COSE
      EN(2,2)=1.
      EN(3,2)=+DEPS
      EN(1,3)=-DPSI*SINE
      EN(2,3)=-DEPS
      EN(3,3)=1.
  200 IF (ABSF(D-DSA)-0.015625) 300,210,210
  210   DSA=D
  220 IMAT=1
D 230 T=DAJUL/36525.0
D 250 A(1,1)=1.-.29697E-3*T*T-.13E-6*T**3
D     A(1,2)=-.0223498*T-.676E-5*T*T+.221E-5*T**3
D     A(2,1)=-A(1,2)
D     A(1,3)=-.00971711*T+.207E-5*T*T+.96E-6*T**3
D     A(3,1)=-A(1,3)
D     A(2,2)=1.-.24976E-3*T*T-.15E-6*T**3
D     A(2,3)=-.10859E-3*T*T-.3E-7*T**3
D     A(3,2)=+A(2,3)
D     A(3,3)=1.-.4721E-4*T*T+.2E-7*T**3
  300 IF (IMAT) 310,400,310
  310 DO   380   I=1,3
  320 DO   370   J=1,3
D 330 ENA(I,J)=0.
  340 DO   360   K=1,3
D 350 ENA(I,J)=ENA(I,J)+EN(I,K)*A(K,J)
  360 CONTINUE
  370 CONTINUE
  380 CONTINUE
      IMAT=0
  400 IF (INVERT) 500,410,500
  410 DO   460   I=1,3
D 420 DVO(I)=0.
  430 DO   450   J=1,3
D 440 DVO(I)=DVO(I)+ENA(I,J)*DVN(J)
  450 CONTINUE
  460 CONTINUE
      GO TO 600
CONTINUED
```

```
  500 DO  550    I=1,3
D 510 DVO(I)=0.
  520 DO   540   J=1,3
D530  DVO(I)=DVO(I)+ENA(J,I)*DVN(J)
  540 CONTINUE
  550 CONTINUE
  600 DO   610   I=1,3
  610 VO(I)=DVO(I)
      RETURN
      END
```

```
*      FORTRAN
CSHAD1  ROUTINE - COMPUTES SHADOW ENTRY AND EXIT POINTS
C   USED WITH N-BODY PROGRAM
C   REM= VEHICLE POSITION (INPUT)
C   VME= VEHICLE VELOCITY (INPUT)
C   TFF= TIME OF FREEFALL (INPUT)
C   DT=  TIME STEP (INPUT)
C   DAJUL= TIME IN DAYS SINCE 1950.0 (INPUT)
C   RT= POSITION AT EDGE OF SHADOW (OUTPUT)
C   VT= VELOCITY AT EDGE OF SHADOW (OUTPUT)
C   T=   TIME OF FREEFALL AT EDGE OF SHADOW  (OUTPUT)
C   ISHAD=0 MEANS COMING INTO SHADOW, ISHAD=1 MEANS COMING OUT OF SHADOW (OUTPUT)
C   KPRNT= 3 OR GREATER FOR DEBUG PRINT (INPUT)
CKCYCL - INTEGRATION CYCLE
C   KCBODY= CENTRAL BODY
      SUBROUTINE SHAD1(REM,VME,TFF,DT,DAJUL,RT,VT,T,ISHAD,KPRNT,KCYCL,
     1KCBODY ,C2, GM)
      DIMENSION  RSUN(3),URSUN(3),REM(3),RRCRS(3),VSUN(3),RMOON(3)
      DIMENSION REM1(3),VME1(3),RT(3), VT(3),VME(3)
      CALL EGEN (4,DAJUL,RSUN,VSUN,KPRNT)
      Ax=6378.165
      Bx=6356.7837
      IF (KCBODY-6) 8,5,8
    5 CALL EGEN (6,DAJUL, RMOON, VSUN, KPRNT)
      DO 7 I=1,3
    7 RSUN(I)=RSUN(I) + RMOON(I)
      Ax=1738.0575
      Bx=1738.0575
      C2=0.0
    8 CONTINUE
      TANALF =695621.73 / MAGF (RSUN,3)
      CALL UNIT (RSUN,URSUN,3)
      CALL VCRS (REM,URSUN,RRCRS)
      RSINTH= MAGF(RRCRS,3)
      CALL VCRS(URSUN,RRCRS,RRCRS)
      CALL RT 1950(RRCRS,0,DAJUL,RRCRS)
      CALL LATLON(RRCRS,BLAT,FLON)
      CALL OBR(Ax, Bx,BLAT,RS)
      B= RS- VDOT(REM,URSUN,3) * TANALF
      RDOTRS= VDOT(REM,URSUN,3)
      IF(KPRNT-3)30,40,40
   40 CONTINUE
      WOT 6, 105, RSINTH,B,RDOTRS
  105 FORMAT (//17H RSINTH,B,RDOTRS= 3E15.8)
   30 SMISS =RSINTH -B
      IF (RDOTRS) 21,21,25
   21 SMISS=1.
   25 CONTINUE
      IF(KCYCL-1)15,12,15
   12 SMISS1= SMISS
   15 CONTINUE
      IF (SMISS) 35,75,45
   35 IF (SMISS1)95,75,75
   45 IF (SMISS1)75,75,95
   75 CONTINUE
      CALL INTRP ( REM,VME, REM1,VME1, DT,TFF, RSINTH,  RSINT1,URSUN,
     1T, RT,VT, TANALF,RS,KPRNT,C2, GM,Ax)
      IF (SMISS1) 77,79,79
CONTINUED
```

```
   77 ISHAD=1
      GO TO 83
   79 ISHAD=0
   83 CONTINUE
      T=T/3600.
      IF(KPRNT-2)95,84,84
   84 CONTINUE
      IF (ISHAD)215,200,215
  200 WOT 6,205
  205 FORMAT ( // 54X,14H ENTER SHADOW  )
      GO TO 310
  215 WOT 6,210
  210 FORMAT ( // 55X,13H EXIT SHADOW  )
  310 WOT 6,305
  305 FORMAT ( /20X,81H POSITION AND VELOCITY IN KM-KM/SEC (EQUAT. COOR.
     1 1950.0) TIME FROM INJ. IN HOURS )
      WOT 6,85,RT,VT,T
   85 FORMAT ( /3H R= 3E15.8, 3H V=3E15.8, 3H T=E15.8 )
   95 CONTINUE
      SMISS1= SMISS
      RSINT1= RSINTH
      DO 97 I=1,3
      REM1(I)=REM(I)
   97 VME1(I)= VME(I)
      RETURN
      END
```

```
*     FORTRAN
CVANGLE ROUTINE - COMPUTES THE ANGLE BETWEEN TWO VECTORS
C       R1      - INPUT VECTOR 1
C       R2      - INPUT VECTOR 2
C       ANG     - OUTPUT ANGLE
      SUBROUTINE VANGLE (R1,R2,ANG)
      DIMENSION R1(3),R2(3),RCROSS(3)
      R1MAG=MAGF(R1,3)
      R2MAG=MAGF(R2,3)
      COSA= VDOT(R1,R2,3)/(R1MAG*R2MAG)
      CALL VCRS(R1,R2,RCROSS)
      SINA= MAGF(RCROSS,3)/(R1MAG*R2MAG)
      IF (RCROSS(3))15,25,25
   15 SINA=-SINA
   25 CONTINUE
      ANG=ANGLE(SINA,COSA)
      RETURN
      END
```

```
*      FORTRAN
CSHAD2  ROUTINE
C COMPUTES ENTRY TO AND EXIT FROM SHADOW,GIVEN R AND V OF INJECTION INTO ORBIT
C NOTE - REQUIERS  TWO PASSES THROUGH SUBROUTINE WITH EPHEMERIS UPDATED BY TMSHD
C  RINJ = INJECTION POSITION
C  VINJ = INJECTION VELOCITY
C  RSUN = POSITION OF PLANET RELATIVE  TO SUN
C  URSUN= UNIT VECTOR POSITION OF PLANET RELATIVE TO SUN
C  GM   = MASS OF PLANET TIMES GRAV. CONST.
C  RS   = RADIUS OF PLANET
C  REX = POSITION  EXIT FROM SHADOW       (OUTPUT)
C  VEX = VELOCITY  EXIT FROM SHADOW       (OUTPUT)
C  TEX = TIME OF EXIT FROM SHADOW -IN SEC. FROM INJ. INTO ORBIT (OUTPUT)
C  REN = POSITION ENTRY INTO SHADOW       (OUTPUT)
C  VEN = VELOCITY ENTRY INTO SHADOW       (OUTPUT)
C  TEN = TIME OF ENTRY INTO SHADOW - IN SEC. FROM INJ. INTO ORBIT (OUTPUT)
C  NOSOL =NO SOLUTION FLAG  -ZERO IS NO SOLUTION  (OUTPUT)
C  KPRNT = DEBUG PRINT FLAG (4 OR GREATER FOR PRINT)
C  TMSHD = TIME FROM INJECTION TO CENTER OF SHADOW -USED TO UPDATE EPHEMERIS
C  COMPUTATION  (IF TMSHD IS ZERO THERE IS NO SHADOW)
C  IBOD = BODY CAUSING SHADOW  (1=EARTH, 2=MOON)
      SUBROUTINE SHAD2 ( RINJ, VINJ,   RSUN, GM, RS, REX,VEX,TEX, REN,
     1VEN,TEN, NOSOL,KPRNT,TMSHD ,IBOD)
      DIMENSION  ONEH(3), URSUN(3),ONEC(3), R(3),V(3), REX(3),VEX(3),
     1REN(3),VEN(3) ,URINJ(3),UVINJ(3)
      CALL VCRS (RINJ,VINJ,ONEH )
      CALL UNIT (RSUN,URSUN,3 )
      CALL VCRS ( URSUN,ONEH,ONEC )
      CALL UNIT ( ONEC,ONEC,3)
      CALL VCRS ( ONEH,ONEC, R )
      CALL VANGLE  ( RINJ,R, DELV)
      IF (ONEH(3)) 5,7,7
    5 DELV= 6.2831853-DELV
    7 CONTINUE
      CALL ORBEL (RINJ,VINJ, GM,A,E ,P, KORB )
      IF (IBOD-1) 122,115,122
  115 CONTINUE
C     IF DELV GREATER THAN 180 DEG.,TRAJ. NOT IN SHADOW
      IF (DELV-3.1415926) 12,12,9
    9 IF (DELV -5.4977) 11,11,10
   10 DELV=DELV-6.2831853
      GO TO 12
  122 CONTINUE
C     IF DELV GREATER THAN  90 DEG.,TRAJ. NOT IN SHADOW
      IF (DELV - 1.5707963) 12,12,11
   11 TMSHD =0.
      GO TO 35
   12 CONTINUE
C    FLY DELV FROM RINJ TO ONER ,COMPUTE R
      CALL AIMP (RINJ,VINJ,KORB,DELV,GM,A,E, P,R, V, NREVS,NOSOL,TMSHD)
      IF (IFG) 62,52,62
   52 IFG=1
      GO TO 35
   62 IFG=0
      CALL UNIT(R    ,URINJ,3)
      CALL UNIT(V    ,UVINJ,3)
      CALL AZGAM(URINJ,UVINJ, AZ,GAMMA )
      TANALF = 695621.73 / MAGF (RSUN,3)
```

```
      B= RS- VDOT(R,URSUN,3) * TANALF
      CALL  VCRS(R,URSUN,ONEC )
      RC= MAGF (ONEC,3)
      D= B* SINF(ACOSF(RC/B))
      IF (KPRNT- 4 ) 76,72,72
   72 CONTINUE
      WOT 6,511, D,B,RC,DELV
  511 FORMAT ( // 13H D,B,RC,DELV= 4E15.8)
   76 CONTINUE
      IF (RC-B) 79,11,11
   79 CONTINUE
      RMAG= MAGF(R,3)
      ADD= D-D*COSF(GAMMA)
      ADD= SIGNF(ADD,GAMMA )
      BETA1=-ATANF ((D+ADD)/ RMAG)
      BETA2=ATANF((D-ADD)/ RMAG)
      DO 25 I=1,2
C  FLY BETA1 AND BETA2 FROM R TO COMPUTE ENTRY AND EXIT POINTS
      CALL  ITSHAD  (R,V, GM, BETA1,RS,URSUN,TANALF,REX ,VEX , TEX ,
     1NOSOL ,KPRNT)
      TEX=TMSHD+TEX
      IF (KPRNT-4)14,13,13
   13 WOT 6,515,REX,VEX,TEX
  515 FORMAT(// 12H SHAD-R,V,T= 7E15.8)
   14 CONTINUE
      IF (I-1) 19,15,19
   15 BETA1=BETA2
      TEN =TEX
      DO 17 J=1,3
      REN(J)=REX(J)
   17 VEN(J)=VEX(J)
   19 CONTINUE
   25 CONTINUE
   35 CONTINUE
      RETURN
      END
```

REFERENCES

1. R. N. Setter and F. M. Chadwick, The N-Body Space Program (VESS), GD|A-DBB64-013, 30 March 1964.

2. F. F. Lisenbe and W. R. Brown, Two-Body Program (VESS), GD|A-63-1073, 5 December 1963.

3. R. N. Setter and F. F. Lisenbe, Lunar Targeting Program II, GD|A-63-0910, 15 October 1963.

4. W. R. Brown, Ephemeris Generator Program (VESS), GD|A-DBB64-017, 30 March 1964.